Department of
Computer Science

UNIVERSITY OF
BATH

Technical Report

PhD. Dissertation: Authentication, Pre-Handoff and Handoff in Pure MANET

Sirapat Boonkrong

Technical Report 2006-14 December 2006

ISSN 1740-9497

Contact Address:
Department of Computer Science
University of Bath
Bath, BA2 7AY
United Kingdom
URL: http://www.cs.bath.ac.uk

ISSN 1740-9497

Authentication, Pre-Handoff and Handoff in Pure MANET

submitted by

Sirapat Boonkrong

for the degree of Doctor of Philosophy

of the

University of Bath

2006

COPYRIGHT

Signature of Author . . . .

Sirapat Boonkrong

Acknowledgements

First of all, I would like to thank Dr. Russell Bradford for his supervision, for all those discussions and for meeting with me every week during my PhD. Without his help and guidance, this thesis would not be possible. I would also like to thank Prof. John Fitch, my second supervisor, for sparing his valuable time to proof-read my thesis and providing useful suggestions. Next is Dr. Julian Padget. I would like to thank him for providing papers and journals earlier on in my PhD. Some of those papers and journals actually inspired me to do what I have done.

I have Dr. Dan Richardson and Dr. Ning Zhang to thank for being my examiners and accepting this thesis. I really did enjoy the viva.

I thank everyone in the lab for keeping me entertained whenever I needed, especially Manu, Adam, Woody, Pete, Matt, Tom, Mark, Jonty and Bill.

Next, I would like to thank everyone in my family, especially my cousins, for their support, for believing in me and making me laugh. P'Nui, P'Kae, P'Noom, P'Golf, P'Noo, P'Nan, P'Nart, Note and Ploy, I thank you all. Kai, my brother, I thank you most for being someone I can talk to about football, rugby and some other stuff. Thanks for helping me take my mind off the work whenever I needed. Thanks for helping me relax whenever I did go back to Thailand.

Most importantly, I would like to thank my mum and dad for their financial support. I would like to thank them for always believing in me, for giving me words of encouragement whenever I was a bit down and for supporting me all the way. Thanks very much, mum. Thanks very much, dad.

Abstract

A mobile ad hoc network (MANET) is a relatively new networking paradigm. Mobile ad hoc networks are more exposed to security threats due to the lack of physical security. That is why the introduction of security protocols, such as authentication and secure handoff protocols, is necessary. This is especially the case for private local mobile ad hoc networks. This thesis is a progress towards producing a secure and efficient authentication protocol as well as a secure and efficient handoff protocol for private local mobile ad hoc networks.

Our investigations show that there are performance and security problems with the existing authentication and keying mechanisms. We design and develop an authentication protocol, which mitigates those problems, using a combination of well-known cryptographic tools of RSA and Diffie-Hellman. The *ns2* simulations demonstrate that authentication and key establishment between two mobile nodes can be accomplished in just four messages, and in approximately ten milliseconds. The protocol has been analysed and proved secure according to the BCK and CK approaches, and the GNY logic.

The pre-handoff protocol has been designed so that mobile nodes will hold the necessary information essential for achieving fast handoffs. This stage is an extension to the normal exchange of *Hello* messages among mobile nodes, which occurs whenever there is a change in network topology. The *multipoint relay* (MPR) algorithm is also integrated into the protocol to improve the efficiency.

One feature of MANETs is the topological instability. Currently, no handoff protocols for pure MANETs are available. We, therefore, introduce an efficient and secure handoff protocol, which is suitable for the movement of any mobile node within that domain, i.e. *micro-mobility.* Our approach again uses the combination of RSA and Diffie-Hellman protocols. We show that a handoff can be accomplished in just five messages, and without relying upon any third parties. The ns2 simulations demonstrate that, on average, it takes less than twenty milliseconds to complete the handoff process, which is more than twice as fast as the time recommended by the ITU. The protocol has also been proved secure and correct according to the BCK and CK formal models, and the GNY analysis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The aim of this research is to design and implement a secure and efficient authentication and session key establishment mechanism, and a secure and efficient handoff protocol (micro-mobility management only) for private local mobile ad hoc networks.

Wireless networking has become commercially and scientifically more and more popular. Wireless ad hoc networks are gaining popularity in recent years due to their mobility, flexibility and ease of deployment. A mobile ad hoc wireless network is a network without any central authority, e.g. a central server. The network is formed by a number of machines (or mobile nodes), which can communicate *directly* with their neighbours by using radio links. If a source node and a destination node are not neighbours, network packets are forwarded from the source to destination by intermediate nodes within the network. This means that each mobile station acts as both a machine and a router.

Two usual objectives of security protocols are to guarantee authenticity and secrecy of communications [Aba99]. Without any security protocols, mobile ad hoc networks can be vulnerable to various kinds of attacks - passive and active attacks [DLRS02] - which can be summarised as follows. No authentication: An attacker could enter and leave the network, and send forged network packets. No encryption: An attacker could "listen" to all the packets within the network. Denial of service attacks, which could be a result of the lack of authentication, are possible. An attacker could re-route the network messages that are sent to a particular node. An attacker could advertise himself as a good route to get all the traffic sent to him. An attacker could replay the sniffed packets. An attacker could masquerade as another node and pretend to be part of the network. More security problems in ad hoc networks can be found in [HBC01].

Since mobile nodes have to rely on each other to forward network packets to destinations, several questions have to be raised. Which user/machine do you allow to join the network? How can you trust a routing node to send the packets to the destinations correctly? How do you know that the node will not send any forged packets? It can be seen that these questions are the ones regarding trust. This is where authentication comes in. More detail on trust can be found in [Den93, PM04]. Many authentication and key agreement protocols [LABW92, BD95, AG00, AST98, Hie00, Sal02, STW97, ZH99, KKA03, BB03, VA00] for ad hoc wireless networks have been proposed. There have also been a lot of interests in the study of secure routing protocols [SGLA96, Che97, PH03, PH02, BT01]. Secure routing protocols concentrate more on the route and topology discovery process, i.e. when a node wishes to find a path to forward a packet, the secure route discovery process will identify the safest path on which the message will be sent. Moreover, the nature of wireless networks is that a mobile node is allowed to move freely within the network. As a result, there has been some interest in handoff or roaming protocols [Per96, MSA03, PC02b, PC02a, CP96, TLP99, PC03, ABABD03, VA00]. However, none of those appear to be suited for a *pure* mobile ad hoc environment. By “pure” mobile ad hoc networks, we mean networks that consist of mobile devices communicating with one another only (no other wired networks or components such as access points or home and foreign agents are parts of the network).

In order to have a secure network, security protocols have to be able to support the following security properties, according to [SA99, ZH99]: *confidentiality*, *integrity*, *authenticity*, *availability* and *non-repudiation*. Confidentiality is to ensure that the information that is being passed is not exposed. The mechanism that protects the confidentiality is simply an encryption process. Integrity makes sure that messages are not corrupted or altered by any malicious nodes. The available mechanisms that help protect the integrity of a message are, for example, message authentication code (MAC) [KBC97]. Authenticity assures mobile nodes that the party they are communicating with is really who that party says he is. Availability ensures that the services from or within the network are available, i.e. this attribute protects the network from such attacks as denial of service attacks. Lastly, non-repudiation states that it is not possible for the origin of a network packet to deny that he has sent the message.

Authentication and handoff processes are two closely related aspects of wireless networks, and are two of the most important issues to consider. The objectives of the work will be to provide an authentication protocol, so that only authorised nodes are allowed to enter and join a mobile ad hoc network; and to provide a secure roaming protocol, so that a mobile station can move freely and securely within a local network. Apart from trying to overcome the various vulnerabilities, there are other aspects of mobile ad hoc networks that we have to consider. They include bandwidth limitations, computational power limitations, battery life limitations and physical security limitations. Therefore, when designing the protocols, we also have to deal with efficiency in addition to security.

The purpose of my thesis is to provide an authentication protocol and a handoff protocol that are more suitable for private local (pure) mobile ad hoc networks. We will show that the existing protocols do not conform to the limitations and restrictions of pure mobile ad hoc networks. We will then propose an authentication protocol, a pre-handoff stage and a handoff protocol which can be applied within private local mobile ad hoc networks.

1.2 Main contributions

The first contribution of this thesis is the examinations and analyses of the existing authentication and keying mechanisms that are currently employed in mobile ad hoc networks, namely ID-based cryptography [Sha84, KKA03, BF01, Sta03], threshold cryptography [Gem97, KKA03, ZH99, Sta03], cluster based authentication [VA00], zero-knowledge based authentication [WYOP94], resurrecting duckling [SA99, Sta01] and others such as location-limited authentication [BB03] and secure spontaneous interaction [KZ04]. We shall argue that these protocols are not practical to be employed within pure mobile ad hoc networks, because they do not take into account the constraints of this type of wireless networks.

Secondly, we will examine and analyse the existing handoff protocols that are currently available. They include Mobile IP [Per96], IEEE802.11 MAC layer handoff process [MSA03], various fast handoff schemes [PC02b, PC02a, CP96, TLP99, PC03], the Sabino System [ABABD03] and [VA00]. We shall argue that these protocols are primarily designed for infrastructure wireless networks, therefore, they are not suited for pure mobile ad hoc networks.

The main contribution of this thesis is the introduction of an authentication protocol, a pre-handoff protocol and a handoff protocol that are applicable and suited for private local mobile ad hoc networks. We design the protocols, analyse the security and prove the correctness of each of the protocols. It will then be shown that due to the performance of the authentication and handoff mechanisms, these proposed protocols are more suitable for employing in pure mobile ad hoc networks than the currently existing ones.

We shall show that by applying two well known cryptographic tools - RSA public-key cryptosystem and Diffie-Hellman key agreement method - only four messages are needed in order to complete the authentication and key establishment. Furthermore, it will be demonstrated that the authentication protocol can be carried out and a new pairwise key can be established in a very short time, between two participating mobile nodes. We will also explain that the reasons that our protocol is more suitable to mobile ad hoc networks than the existing ones are as follows. The protocol eliminates the problem of single point of failure. The protocol does not require any human interactions. Authentication can be accomplished even if mobile nodes are not in the line-of-sight. In addition, as a result of a protocol run, mobile nodes do not have to rely on any other parties for communications.

The pre-handoff protocol will be developed in order to get mobile nodes ready for any handoff process that may take place in the future. When there is a change in network topology, e.g. a new mobile node has just joined the network, most routing protocols provide a mechanism for mobile nodes to update their routing information and topology tables. That mechanism is the exchange of *Hello* messages among the mobile nodes. In this stage, we have added an extension to this mechanism, so that necessary information for a handoff can be passed around the mobile nodes. Moreover, we integrate an efficient flooding technique known as Multipoint Relay (MPR) algorithm, so that the use of network bandwidth becomes more efficient.

For the handoff protocol, we will show that in general a handoff and disassociation can be accomplished in just five messages. It is also possible to complete a handoff quickly enough without losing the connection, even if the roaming mobile node travels at a high speed, e.g. the speed of a train. In addition, we shall demonstrate that any mobile node can carry out the protocol without relying on any extra pieces of hardware, which is one of the reasons that our handoff protocol is more suited for pure mobile ad hoc networks.

Some of the results of this thesis have been accepted for publication previously. The authentication protocol was published in the proceedings of the first Thailand Computer Science Conference 2004 (ThCSC 2004). The pre-handoff stage and the handoff protocol were accepted to appear in the proceedings of the third International Workshop in Wireless Security Technologies 2005 (IWWST 2005).

1.3 Structure of the thesis

This thesis is organised as follows. First, chapter 2 will provide the background knowledge needed to understand the thesis. This will include the general description and characteristics of mobile ad hoc networks, the descriptions of RSA public cryptosystem and Diffie-Hellman key agreement protocol. They are then followed by the overviews of the BCK, CK models and GNY protocol, all of which will be used for the analysis and proof purposes.

In chapter 3, we provide the overviews of the existing authentication and keying protocols which are currently employed in mobile ad hoc networks. Even though there are no handoff protocols specifically for pure mobile ad hoc networks, we shall give the overviews of the existing handoff protocols that are available for infrastructure wireless networks. In this chapter, we also examine the existing authentication and handoff protocols, and evaluate how they may or may not be suited for pure mobile ad hoc networks. We shall argue that these techniques are not satisfactory because they are unable to overcome some of the restrictions and limitations of mobile ad hoc networks.

In chapter 4, we propose and design an authentication protocol, a pre-handoff stage and a handoff protocol, all of which will be the solutions to the physical constraints of mobile ad hoc networks and the problems of the existing protocols. For the authentication and handoff protocols, we provide the security analyses using the BCK and CK formal models as well as the proofs of correctness using the GNY logic. The main purpose of the analyses and proofs is to emphasise the security of the protocols.

Chapter 5 describes and explains the simulations that have been run in order to test the authentication, pre-handoff and handoff protocols. In general, we simulate two different settings - outdoor environment and indoor (office) environment.

The results of the simulations are provided in chapter 6. The main purpose of running the simulations is so that we can demonstrate and examine the performances of the three protocols - authentication, pre-handoff and handoff. This chapter also provides the discussions and evaluations for the authentication protocol, the pre-handoff protocol and the handoff protocol. We shall see whether our protocols satisfy the design requirements, and whether or not they are able to overcome the problems that the existing protocols have.

Chapter 7 provides the conclusion of the thesis as well as possible directions for future research.

The notations and postulates of the GNY logic are given in appendix A. The diagrams of the larger network layouts used in the simulations are found in appendix B.

Chapter 2

Background Knowledge

This chapter explains the necessary background knowledge that is useful or has been applied to this research. We start with a description of mobile ad hoc networks, which is the focus of this research. We then give brief descriptions of the RSA public-key cryptosystem and the Diffie-Hellman key agreement protocol. A modular approach to the design and analysis of authentication and key exchange protocols, namely the BCK approach, is explained in detail, together with the method for the analysis of key-exchange protocols and their use for building secure channels, or the CK method. These are the formal models that we followed during the design and analysis of the authentication and handoff protocols. GNY analysis will also be described here.

2.1 What is a mobile ad hoc network?

A mobile ad hoc network (MANET) [CM99] or an infrastructureless wireless network is a network formed, without a central authority (such as a server), by the collaboration of a number of mobile nodes. These machines can communicate with their neighbours *directly* by using radio links. Network packets are relayed from one machine (source) to another (destination) by other nodes within the network, on behalf of the source station, if and when the source and destination mobile nodes are not within each other's range. This means that each station acts as both a machine and a router.

Routing is defined as a function which determines the path from a source to a destination for the traffic flow. Routing protocols for mobile ad hoc networks are divided into three categories, on-demand, table-driven and hybrid. On-demand routing protocols create routes by initiating a route discovery mechanism when the routes are required by the source node. This type of protocols includes, but is not limited to, Ad Hoc On Demand Distance Vector (AODV) [PR99], Temporally-Ordered Routing Algorithm (TORA) [PC97], Dynamic Source Routing (DSR) [JM96], Associativity-Based Routing (ABR) [Toh97] and Signal Stability Routing (SSR) [DRWT97].

Table-driven routing protocols maintain up-to-date routing information from each node to every other node in the network. These routing protocols require each node to maintain one or more tables, such as a topology table and a routing table, to store necessary routing information. When there are changes in the network, the nodes respond to them by propagating updates throughout the network. Examples of the table-driven routing protocols are Dynamic Destination-Sequenced Distance Vector (DSDV) [PB94], Wireless Routing Protocol (WRP) [MGLA96], Clustered Gateway Switch Routing (CGSR) [CCG97], Zone-based Hierarchical Link State (ZHLS) [JNL99a], Fisheye State Routing (FSR) [PGC00] and Optimized Link State Routing Protocol (OLSR) [CJ03].

Hybrid routing protocols combine the features of both on-demand and table-driven routing protocols. Routes are maintained in the same way as the table-driven protocols by mobile nodes that are close to one another. For nodes that are further away, paths are discovered by using a route discovery strategy which is defined as part of the on-demand routing protocols. Examples of the hybrid routing protocols include Zone Routing Protocol (ZRP) [HP99], Zone-based Hierarchical Link State (ZHLS) [JNL99b], Scalable Location Update Routing Protocol (SLURP) [WS01] and Distributed Dynamic Routing (DDR) [NLB00].

Another main characteristic of an ad hoc network is that there is always topological instability within the network due to the ability of the mobile nodes to move freely from one position to another. That is why a suitable routing protocol needs to be applied to the network in order to discover and maintain routes among the nodes in the network.

The origin of ad hoc networks and reviews of some of the routing protocols can be found in [FJL00] and [RT99, AWD04] respectively.

Wireless ad hoc networks have gained popularity in recent years due to their mobility, flexibility and ease of deployment. There are, however, several constraints on this type of wireless networks. They include bandwidth limitations, computational power limitations, battery life limitations and physical security limitations. In comparison with hardwired networks, the bandwidth of a wireless network is considerably smaller. For example, for an 802.11b wireless network, the bandwidth is between five and eleven megabits per second, whereas gigabit Ethernet is now widely available. Other factors that affect the bandwidth in wireless networks include signal fading and signal interference. This suggests that when developing a security protocol, we would want it to generate as few messages as possible, and each of the messages should be as short as possible. Moreover, a mobile station is typically small, e.g. a hand-held computer or a laptop, whose computational power and battery life are not as great as a conventional desktop computer. This implies that security protocols should not consume much of the resources that are available. Mobile ad hoc networks also have limited physical security. They are more prone to physical security threats than their hardwired counterparts. In wireless networks, there is an increased possibility of eavesdropping and spoofing. Therefore, when designing a protocol, we have to take into account the constraints of mobile ad hoc networks as well as the fact that they are more vulnerable to attacks than the wired networks.

2.2 RSA cryptosystem

A public-key cryptosystem, specifically RSA, will be applied in our proposed authentication and handoff protocols. RSA will serve two purposes when used in our protocols. Firstly, we will use it for encryption during the exchange of messages before any pairwise key is established. Secondly, RSA will be used for authenticity and identification purposes. For example, when a message is sent, the sender will "sign" the message, so that it is possible for the receiver to be sure that the message is from the expected source. Alternatively, A could send a challenge encrypted with B's public key. B could then decrypt the challenge with his private key, and send the decrypted challenge back to A. A would be able to know that B is holding the correct private key by checking the received decrypted challenge.

RSA is a public-key cryptosystem in which each user has a private key and a public key. As the names suggest, the public key is freely available whereas the private key remains secret to the owner. In public-key cryptosystems, encryption is done with the public key and decryption is done with the private key. Public-key cryptosystems also offer digital signatures, which can be used by the user to *sign* a message. A digital signature can be thought of as a "stamp", which is unique to each user and is difficult to forge. The signature also assures that any changes made to the data that has been signed cannot go undetected. Therefore, digital signatures can be used to prove the origin as well as the authenticity of the message.

For example, in the RSA system, Alice and Bob each picks a public and private key pair. They make the public keys freely available (i.e. Bob knows Alice's public key and Alice knows Bob's public key) and keep the private keys to themselves. When Alice sends a message to Bob, she encrypts it using Bob's public key and signs the message with her own private key. When Bob receives the message, he decrypts it using his private key, and verifies the signature (authenticity/origin) using Alice's public key. The digital signature helps Bob ensure that the message really comes from Alice.

A full description of the RSA cryptosystem can be seen in [RSA78, MvOV96].

2.3 Diffie-Hellman key agreement

The Diffie-Hellman key agreement protocol [DH76] allows two users to establish a new shared secret key to use between them, without any prior secrets. This key agreement method will be integrated into both our authentication and handoff protocols, so that the two communicating parties can establish a new shared key, which can be used for encryption and decryption in further exchange of messages. The protocol works as follows.

Alice and Bob share a large prime p and generator g (g has the following property: for every number n between 1 and $p-1$ inclusive, there is a power k of g such that $n = g^k \bmod p$). Alice chooses a large integer x, which becomes her secret component, and computes $X = g^x \bmod p$. Bob chooses a large integer y, which becomes his secret component, and computes $Y = g^y \bmod p$. Alice sends X to Bob, who then calculates $K_{ab} = (g^x)^y \bmod p$. Bob sends Y to Alice, who then computes $K_{ab} = (g^y)^x \bmod p$. Alice and Bob now have the same shared secret key, K_{ab}.

The Diffie-Hellman key agreement is difficult to break due to its dependence upon the discrete logarithm problem. That is, given $X = g^x \bmod p$ and $Y = g^y \bmod p$, it is computationally infeasible to calculate either x or y, provided that p is a sufficiently large prime. Having said that, the Diffie-Hellman key agreement is not totally secure as it is vulnerable to man-in-the-middle attacks.

A full description of the Diffie-Hellman protocol can be seen in [DH76, MvOV96].

2.4 BCK and CK analysis methods

There are two approaches to the protocol analysis. They are simulation-based and indistinguishability-based. The BCK and CK methods fall into the latter category. The reason that we have chosen the BCK and CK analysis methods is because by following their approach, we are able to gain the benefit of simpler analysis and easier-to-write proofs of security.

The authentication and secure communication problems deal with preventing adversaries from controlling the communication links used by legitimate parties. The adversaries may modify the contents of messages, they may delete messages and they may replay messages to any mobile nodes within a network. The goal of security analysis is to make sure that protocols become less vulnerable to the mentioned attacks. The BCK and CK methods offer attractive approaches to security analysis. That is, they provide a modular treatment for analysing and proving the security of cryptographic protocols. A modular approach is more efficient and less time consuming than methods that evaluate a whole system at once. [SZ98] also supports the use of modular methods by saying that the security of a system is determined in terms of that of its components.

A modular approach to the design and analysis of authentication and key exchange protocols was proposed by Bellare, Canetti and Krawczyk [BCK98a, BCK98b]. By using this method during the design and development of a protocol, we can construct a protocol that is secure in an unauthenticated network. Before we begin explaining the main ideas of the BCK model, we will have to define the terms that are presented in [BCK98a, BCK98b].

Suppose there is an adversary who is capable of carrying out such attacks as modifying messages, deleting messages and injecting false messages. The idealised *authenticated-links model* is where the adversary cannot behave as he wishes, i.e. he cannot carry out any attacks. The attacker is restricted to delivering messages faithfully. On the other hand, the *unauthenticated-links model* is where the adversary is not constrained to delivering messages honestly. He can do anything he wishes to the messages, namely modifying them, deleting them and injecting them.

The main component in the BCK modular approach is the concept of *authenticators*. An authenticator takes a protocol that is proved to be secure in the authenticated-links model, and turns it into a protocol (that computationally achieves similar characteristics) that is secure in the unauthenticated-links model. Formally, these authenticators are called *universal compilers*, $\mathcal{C}$. Given a protocol, π, in an authenticated model, a compiler $\mathcal{C}$ transforms it into a protocol, π' in an unauthenticated model. This process can be represented as $\pi' = \mathcal{C}(\pi)$. A more detailed definition of the authenticated-links model, unauthenticated-links model and a compiler are presented in [BCK98a, BCK98b].

The essential idea of the BCK approach is that "we start with solutions that work in a model of idealised authenticated communications and then transform these solutions, via automatic techniques (or *compilers*), into solutions that work in the realistic unauthenticated setting" [BCK98a, BCK98b]. Informally, we design an authentication or a key exchange protocol, and prove that it is secure in the authenticated-links model. We then construct and apply a particular *authenticator* or *compiler* to the designed protocol in order to transform it into a protocol that is secure in the unauthenticated-links model. How do we know that the resultant protocol is a secure protocol in the unauthenticated-links model? The resultant protocol must *emulate* the ideal process or the secure authenticator, where emulation of protocols means that "running π' in an unauthenticated network has the same effect as running π in an authenticated network" [BCK98a, BCK98b].

[BCK98a] and [BCK98b] introduce a signature-based authenticator, λ_{sig}, which will be applied to parts of our protocols during the construction, analyses and proofs. Bellare et al have proved in [BCK98a, BCK98b] that if the signature-based authenticator, λ_{sig}, is applied to a protocol then that protocol will become secure in unauthenticated networks. The signature-based authenticator works as follows.

The protocol λ_{sig} is a two-party protocol. First, party A sends '`message`:m'

to party B. Upon receipt of '`message`:m' B chooses a random value $N_B \in_R \{0,1\}^k$, and sends '`challenge`:m, N_B' to A. Upon receipt of that from B, A sends '`signature`:m, $SIGN_{s_A}(m,\ N_B,\ \mathrm{B})$' to B. Upon receipt of '`signature`:m, $SIGN_{s_A}(m,\ N_B,\ \mathrm{B})$', B accepts m if the signature is successfully verified. Figure 2-1 depicts the exchange of messages described here.

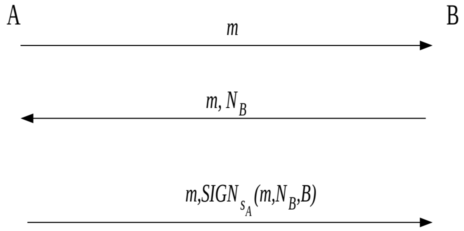


Figure 2-1: Signature-based authenticator

Canetti and Krawczyk introduce the *analysis of key-exchange protocols and their use for building secure channels* [CK01]. In their paper, they apply the concepts of the BCK modular method and explain in more detail the capabilities of attackers, both in the unauthenticated-links model and the authenticated-links model.

In addition to the basic attacking capabilities, mentioned earlier in this section, Canetti and Krawczyk suggest that the attackers are capable of obtaining "secret information stored in the parties' memories via explicit attacks" [CK01]. The attacks are classified into the following categories.

- `Party corruption`: When a party is corrupted, the attacker learns all the internal memory of that party, including the long-term secrets (such as private keys or shared secret) and short-term secrets (such as session keys).

- `Session-key query`: The attacker provides a party's name and a session identifier of a completed session at that party, and receives the value of the key generated by that session. This attack provides the information on specific session keys which may result from such events as break-ins, cryptanalysis and careless disposal of keys.

- `Session-state reveal`: The attacker provides a party's name and a session identifier of an incomplete session at that party, and receives the state of that session. For example, the information revealed via this attack may be some components that are used for generating a session key. Note that the attacker does not learn anything about long-term secrets via this attack.

If a session is subject to any of the above attacks, then the session is called *locally exposed*. If a session or its matching session is locally exposed, then we call the session *exposed*.

Not only do Canetti and Krawczyk explain the possible attacks, they also introduce a notion of *session-key security*. They claim that for a key-exchange protocol to be secure, an attacker must not learn anything about the value of the key from interacting with the key-exchange protocol and attacking other sessions and parties. The definition of *SK-security* is as follows.

Canetti and Krawczyk start by defining an "experiment" in which the attacker, $\mathcal{U}$, will be asked to differentiate the real value of the session key from a random value. The attacking capabilities of $\mathcal{U}$ are extended so that $\mathcal{U}$ is allowed to choose a *test-session* among the sessions that are completed and unexposed at the time. We extract the following definition from [CK01].

Let K be the value of the corresponding session-key. We toss a coin, b, $b \xleftarrow{R} \{0,1\}$. If $b = 0$ we provide $\mathcal{U}$ with the value K. Otherwise, we provide $\mathcal{U}$ with a random value r. The attacker $\mathcal{U}$ is now allowed to continue with the regular actions, but is not allowed to expose the test-session. At the end of his run, $\mathcal{U}$ outputs a bit b' as its guess for b. Below is the formal definition, Definition 4 [CK01], of SK-security.

Definition (4 of [CK01]): *A protocol π is called* `SK-secure` *if the following properties hold for any KE-adversary $\mathcal{U}$ in the unauthenticated-links model.*

1. *Protocol π satisfies the property that if two uncorrupted parties complete matching sessions then they both output the same key; and*
2. *the probability that $\mathcal{U}$ guesses correctly the bit b (i.e., outputs $b' = b$) is no more than $1/2$ plus a negligible fraction in the security parameter.*

If the above properties are satisfied for all KE-adversaries in the authenticated-links model then we say that π is `SK-secure` *in the authenticated-links model.*

Another important concept introduced in [CK01] is the notion of *Secure Channels*. A secure channel is a communication link between two parties that, by using a shared key generated by a key-exchange protocol, achieves authentication and secrecy. In [CK01], the authors show that "an SK-secure key-exchange protocol, appropriately combined with secure MAC and symmetric encryption functions, suffices for realising such secure channels" [CK01].

In order to understand what a secure network channel is, Canetti and Krawczyk formalise two important definitions, *Secure network authentication* and *Secure network encryption*.

Network authentication is defined as a two-party protocol, π, in which one party produces an output $m' = (m, f_k(m))$ where m is a message and f is a MAC function, and the receiver outputs (v, ok) as follows. If m' is of the form (m, t) then $ok = 1$ if and only if (i) m is different from all previously received messages in this session (preventing replay attacks), and (ii) (m, t) passes the message authenticity verification. If $ok = 1$ then $v = m$, otherwise $v = null$. A protocol is called a *secure network authentication* protocol if it emulates protocol SMT in the unauthenticated-links model. The protocol SMT is defined in [CK01].

A protocol is considered a *secure network encryption* protocol if an attacker does not learn information on messages that are exchanged during an unexposed session. The concept of the secure network encryption protocol is introduced to overcome the problem of secrecy of communications. The more formal definition of secure network channel protocols can be found in [CK01].

We are now ready to define what is meant by *secure channels*. The following Definition 15 is the formal definition of secure channels presented in [CK01].

Definition (15 of [CK01]): *A protocol is called a secure network channels protocol if it is a secure network encryption protocol and also a secure network authentication protocol.*

In general, by using the BCK modular approach, we can break up a protocol into smaller and simpler protocols. The modular approach simplifies the design and analysis work, and helps building the protocol in ways that are easier to implement and maintain in practice. In [CK01], we gain some necessary knowledge that will help make the analyses and proofs more complete.

2.5 GNY protocol

It is useful and necessary to apply the GNY logic alongside the BCK and CK analysis methods. This is because while the BCK and CK methods make sure that communication channels are secure, the GNY logic provides a way of examining the components of the protocol messages when they are sent and received. In other words, GNY protocol helps us figure out what could potentially go wrong with protocols, so that we could change it. It does not take into account, however, the communication channels [LABW92]. This is where the BCK and CK methods come in. Moreover, one could say that GNY protocol focuses mainly on the authenticity of protocol messages whereas the BCK and CK methods address secrecy. Therefore, the GNY logic is complementary to the BCK and CK methods [GSG99].

The GNY protocol is a formal tool that allows us to analyse cryptographic protocols, step by step according to the rules provided (they can be found in Appendix A). The GNY protocol was developed by Gong, Needham and Yahalom [GNY90] as a new approach based on a similar tool called BAN logic [BAN90].

A protocol, in this context, consists of the exchange of some network messages between two or more principals. A protocol determines what and when messages should be sent, and by which particular principal. Each protocol run is referred to as a *session*.

According to the GNY logic, each principal in a session maintains two sets, which are the *belief* set and the *possession* set. The belief set contains all the beliefs of the principal. The possession set contains everything the principal has received and generated in a particular session.

At the start of a new protocol session, principals have initial sets of beliefs and possessions (i.e. initial assumptions). A principal can obtain new beliefs as a result of receiving new messages, thus enlarging the belief set. The rules provided as parts of the GNY protocol "enable the derivation of new beliefs from current beliefs and incoming messages" [GNY90]. The possession set can be expanded when receiving new messages as well [1].

When analysing a protocol, we begin with a set of initial assumptions. The

[1] "If a belief or a possession is a member of its respective set at any phase of some session, then it is a member of that set at any subsequent phase of that session." [GNY90]

appropriate rules (*Being-told, Possession, Freshness, Recognisability, Message interpretation* and *Jurisdiction* rules, usually in this order) are then applied to each of the received messages. Once the analysis is finished, the protocol should end up with the expected outcomes, e.g. a new shared secret. If in any case the analysis does not terminate, or the protocol does not achieve the expected conclusions, the feature of the GNY protocol is that it allows the user to see at which step the analysis is stuck or potentially can be attacked. In other words, the analysis allows us to realise which essential components are missing from which message, or what attacks are possible at what stage of the protocol. For example, if the analysis is stuck when applying the freshness rule, we know that there is nothing fresh in the received message, which could lead to a replay attack. From this, we should be able to add some necessary elements to that message, so that any potential replay attacks become impossible.

GNY protocol provides a debugging tool, and may help protocol designers get rid of unnecessary elements. For more details about the rules of the GNY protocol and the logic of GNY itself, the reader is referred to [GNY90], while [MSNN94] presents some remarks on the GNY logic.

Summary

Here we have presented the definition and the important characteristics of a mobile ad hoc network, and briefly explained the RSA public-key cryptosystem and Diffie-Hellman key agreement protocol. We have also summarised the ideas of a modular approach to the design and analysis of authentication and key exchange protocols (the BCK model), the analysis of key exchange protocols (the CK model), and the logic of GNY used for proving the correctness of cryptographic protocols. All of the above are essential concepts for designing, developing and analysing the protocols in this research.

The RSA public-key cryptosystem and the Diffie-Hellman key agreement will be integrated into the proposed authentication and handoff protocols. The RSA will provide encryption, authenticity and identification prior to the establishment of a new pairwise key (which applies the Diffie-Hellman method). The BCK and CK approaches will provide the analysis for the security of the communication channel between two parties, while the GNY protocol will provide the examination of each individual protocol message.

Now we have obtained the necessary background knowledge. In the next chapter, we will give the descriptions and analyses of the protocols (both authentication and handoff protocols) that are related to and were previously developed prior to this research.

Chapter 3

Existing Protocols and Their Problems

This chapter contains descriptions of the work that was previously developed, and is related and relevant to the work done in this research. We have divided it into two categories: authentication and keying mechanisms, and (secure) handoff protocols. Following this, we state the reasons why the existing protocols fail to provide adequate solutions to our problems, namely the security, physical limitations and restrictions of pure mobile ad hoc networks.

3.1 Authentication and keying mechanisms

There are many proposed authentication protocols and keying mechanisms for ad hoc wireless networks. In this section, we will review each of them in turn. The existing protocols that will be described in this section are ID-based cryptography [Sha84, KKA03, BF01, Sta03], threshold cryptography [Gem97, KKA03, ZH99, Sta03], cluster based authentication [VA00], zero-knowledge based authentication [WYOP94], resurrecting duckling [SA99, Sta01] and others such as location-limited protocol [BB03] and secure spontaneous interaction [KZ04].

We will then analyse the authentication and keying mechanisms, and explain the reasons why they fail to overcome the problems of physical limitations of mobile ad hoc networks. When examining the existing protocols, it is important to realise that it is more difficult to design a solution to problems in mobile ad hoc networks than in conventional wired networks due to several factors. Firstly, the nature of a pure ad hoc network is that it has no central authority, such as a server, which is usually largely responsible for authentication and key distribution. Other constraints include bandwidth limitations, computational power limitations, battery life limitations and physical security limitations. Therefore, when designing an authentication protocol, we must keep these physical conditions in mind. The resultant protocol should consume as little resource as possible. In addition, we would like it to be both efficient and, more importantly, secure.

3.1.1 ID-based cryptography

ID-based cryptography was first proposed in [Sha84]. The idea was developed further by Boneh and Franklin [BF01]. The overview of this scheme is provided below.

An ID-based cryptosystem consists of four main algorithms.

1. **Setup** - generates global system parameters and a master-key.
2. **Extract** - uses the master-key to generate the private key corresponding to an arbitrary public key string.
3. **Encrypt** - encrypts messages using the public key ID.
4. **Decrypt** - decrypts messages using the corresponding private key.

The following is an example of ID-based cryptography. If Alice sends Bob a message, she will encrypt the message using Bob's identity, such as his email or his MAC address, as the encryption key. When Bob receives the message, he will have to contact a third party machine known as a *private key generator* in order to get a private key for the decryption. However, Bob will have to authenticate himself to the private key generator before the private key generation process begins. Once Bob has obtained the private key, he will then be able to decrypt the message and read it.

There are advantages gained from using this key distribution scheme. Firstly, it can be seen that public keys do not have to be distributed, because they are simply the identities of the machines or users. This plus point is feasible for ad hoc wireless networks in that no messages would be needed to distribute the keys. Thus, the usage of power and network bandwidth can be reduced.

On the other hand, this cryptosystem is not suitable for mobile ad hoc networks, because it is necessary to involve a third party node (a private key generator) in order to establish a private key. This means that the ad hoc networks would lose their flexibility and scalability. Furthermore, if each user were able to compute his own key, then it would not be possible to have a secure network. This is because of a specific way that a private key is generated by an algorithm that takes user's identity and a random seed as inputs. Shamir confirms that "if user A could compute the secret key that corresponds to the public key 'A', he could also compute the secret keys that correspond to the public keys 'B', 'C' etc. and the scheme would not be secure" [Sha84]. Another problem is that the private key generator could produce an incorrect key and give it to the user. This would prevent the user from decrypting the messages he receives correctly.

Even though there are limitations to the ID-based protocol, Khalili et al have applied the ideas of ID-based cryptography and threshold cryptography to distribute keys in ad hoc networks [KKA03].

3.1.2 Threshold cryptography

Threshold cryptography is presented in [Gem97]. An example of applications for this scheme is key escrow protocols in which two or more third parties (or escrow agents) hold parts of a key or keys. The security provided by the threshold cryptography includes confidentiality, and data integrity and availability in the presence of dishonest shareholders. We describe the concept of the threshold cryptography below.

A *k-of-n* threshold protocol, Gemmell explains, involves *n* nodes known as *shareholders*, possibly one machine called a *dealer* and one called a *combiner*. The shareholders hold pieces of data known as *shares*, which are initially distributed by a dealer. In this particular protocol, any *k* shares can be combined, by a combiner, to generate a secret. This means that, in the *k-of-n* protocol, "no *k-1* shares yield any significant information about the secret" [Gem97].

Threshold cryptography has been applied for securing ad hoc networks by Zhou et al and Khalili et al, and [ZH99, KKA03] present the protocols in detail.

Both papers propose a similar approach for dealing with key distribution. The main difference is that Khalili et al have also integrated ID-based cryptography into their scheme. [KZL$^+$01] and [LL00] also use the concept of threshold cryptography as the basis of their protocols.

After we have studied the threshold protocol and the two schemes proposed in [ZH99, KKA03], we can conclude that the main advantage present in this protocol is that it reduces the problem of single point of failure. In other words, for a *k-of-n* protocol, the network is said to be broken only if k or more machines are compromised by an attacker. However, it does not eliminate the problem of single point of failure altogether. That is, if a combiner or a dealer is compromised, the protocol will fail. Another benefit is that it is possible for a node to obtain its secret as long as k machines are reachable, i.e. this mechanism makes the protocol more flexible when any k of n machines are available.

The disadvantages to this scheme are as follows. Firstly, there are some limitations to mobile ad hoc networks, such as the limited amount of network bandwidth and battery power. That means that we would want to send and receive fewer messages in order to accomplish a task. In this protocol, however, the user will have to contact at least k machines before obtaining a secret key and reading or decrypting his message. There is also a danger of not having k nodes available at any one time. If that happens, the user will have to figure out a way of obtaining the required number of shares. One such way is to move his machine so that more nodes can be reached which can give him enough shares to generate a secret key. This implies that there is a possibility that the user will be spending more time trying to obtain a secret key, and wasting unnecessary computing resources.

There are other problems in [KKA03] which should be pointed out. That is when a new node is joining an existing network, it seems that the protocol could be vulnerable to man-in-the-middle attacks, and malicious (existing) members could provide the newly joined node with a false master public key [1]. Moreover, how can we trust the shareholders to give the user a correct value of his/her share? An adversarial shareholder could lie about his/her value which would prevent a successful reconstruction of the secret key.

[1] [KKA03] states that a master public key is given to all members when they join.

It can be seen that threshold cryptography gives some benefits and drawbacks to the security of mobile ad hoc networks when the protocol is employed.

3.1.3 Cluster based authentication

Cluster based authentication [VA00] is based on the concept of a routing protocol called Clusterhead Gateway Switch Routing. The cluster based architecture, as claimed by Venkatraman and Agrawal, is most suitable for large networks.

The entire network is divided into a number of overlapping clusters, each of which has to elect a cluster head. Before a node joins a network, a *system public key* and a *system private key* are given to the node. This pair of keys is shared by all the nodes in the network. A *cluster key*, which is generated by the cluster head is also given to the node. This cluster key is unique to that particular cluster, and shared by all the nodes in that cluster.

When a node joins a network, a challenge-and-response authentication is carried out, with the system key pair used for mutual authentication between the joining node and the existing member of the network.

If two nodes from different clusters wish to communicate with one another, they will have to establish a session key. When a node wants to start a session with another node from another cluster, it sends a request to the cluster head, who will then generate a set of random prime numbers[2]. These numbers are broadcast to all the nodes in the cluster. Authentication can be achieved by verifying the origin and authenticity of the tags. The full description of the algorithm for the communication between nodes can be found in [VA00].

The authors, Venkatraman and Agrawal, present the advantages and limitations of the cluster based authentication in their paper [VA00]. The advantages include the introduction of mechanisms for packet authentication, preventing replay attacks and ensuring data integrity. On the other hand, the limitations, stated in the paper, are to do with how random prime numbers or tags should be generated. More importantly, though, is the fact that secret keys are not freshly generated for each session.

A further advantage is that the computational cost of the per-packet authen-

[2] These numbers are also known as authentication tags.

tication is reduced by using the authentication tags. The major disadvantage of this scheme is the problem of a single point of failure. It appears that the cluster head plays a very important role in this protocol. Its duties include cluster key generation, creating authentication tags and timestamps, and dealing with some encryption. If the cluster head is compromised, that particular cluster will not be able to function and communicate with the others.

We can see that the cluster based authentication provides several positive features, but the reliance on cluster heads makes it less attractive to mobile ad hoc networks.

3.1.4 Zero-knowledge based authentication

Zero-knowledge protocols are explained with the help of a certain Ali Baba story, *The Strange Cave of Ali Baba*. This story was used to explain the idea of zero-knowledge by Quisquater, Guillou and Annick. We would like to refer the reader to [QGAB89] for the complete story. The Ali Baba story goes like this.

Very long time ago, in the Baghdad bazaar, a thief grabbed a purse from Ali Baba and fled into a cave. Ali Baba ran after him, and discovered that inside the cave there were two passages, left and right. Ali Baba had to choose which passage the thief went into. He decided to go to the left, searched everywhere from the entrance, and found that the left passage ended in a dead end. Ali Baba thought the thief was lucky that he chose the wrong passage. On the next day in the bazaar, another thief stole Ali Baba's basket, and again ran to the same cave. This time Ali Baba decided to go into the right passage. Again the right passage ended in a dead end. Ali Baba thought the thief was lucky once again. The same thing happened for forty days, and Ali Baba did not catch one thief. He thought that there was no way that the thieves were lucky in every occasion. Therefore, he decided to hide under a sack in one of the passages to see how the thieves had escaped him every time. Finally, a thief came in, said the magic words, "open sesame", and to Ali Baba's astonishment, the wall at the dead end opened. Now Ali Baba knew the magic words. He worked and he worked on the words until he was able to change them to his own new magic words.

Ali Baba had recorded the story. He did not say what the new magic words were but he did include some subtle clues about them. The story became an interest to many historians. Finally, Mick Ali came out and said that he knew what the magic words were. He had proved it on television by going into one of the passages (but he did not let anyone come to the dead end with him as he did not want to reveal the secret words) and having a reporter flip a coin. If the coin came up heads, Mick come out on the right. If the coin came up tails, Mick would come out on the left. They experimented forty times and every time Mick came out on the correct side.

By doing what Mick Ali had done, he had achieved his real objective. That is he had shown that it was possible to convince the audience that he knew the secret words, without having to reveal them. This idea would be the basis of the zero-knowledge authentication, which is explained below.

The idea of zero-knowledge authentication and identification was first introduced by Fiat and Shamir in [Odl87]. The concept was developed further and applied by Wulf et al [WYOP94] so that the scheme would become more suitable to the authentication in distributed systems. We will provide the concept of the scheme below. [WYOP94] presents the protocol in more detail.

Zero-knowledge based authentication means that the authentication is carried out without revealing any secret shared by any parties involved in the process. It is a "mechanism that allows the verifier to ask a question to which the answer can be verified without knowing what computations lead to the answer" [WYOP94]. The main components of this protocol are two functions, f and g. They must have the following properties:

$$\begin{gathered} f \text{ and } g \text{ are hard to invert and} \\ g(f(x), f(y)) = f(g(x, y)) \end{gathered}$$

Here is how the zero knowledge authentication is carried out. For example, user A would like to identify himself or herself to user B.

- User A selects a function f, which is only known by him or her. A sends a message to B containing his own identity, an arbitrary value x_0 and the value of $f(x_0)$. The message is denoted as:
  $A \rightarrow B : A, x_0, f(x_0)$
  where x_0 is an arbitrary value.

- Function g is selected by user B to authenticate future requests for service from A. Only B knows g. To authenticate, B generates an arbitrary value, y, and sends a message to A containing the value y and the value of $g(x_0, y)$.
  $B \rightarrow A : y, g(x_0, y)$
- A would then return a message to B containing the values of $f(y)$ and $f(g(x_0, y))$.
  $A \rightarrow B : f(y), f(g(x_0, y))$.
- B calculates $g(f(x_0), f(y))$ and compares that with the received $f(g(x_0, y))$. If the values are the same, B can conclude that A is really who he claims to be.

The authors have shown in their paper [WYOP94] a couple of examples of the functions f and g. The examples include $f(x) = x^2$, $g(x, y) = xy$ and $f(x) = x^a \bmod n$, $g(x, y) = (xy)^b \bmod n$. Furthermore, there are two "critical elements", mentioned by the authors, to this scheme. They are the unpredictability of the function $g(f(x_0), f(y))$, which must only be known by the authenticator (B in this case). The other element is that only the user A knows f, otherwise A could be masqueraded by an attacker. This means that the security of this method relies on the privacy and unpredictability of f and g.

Other authentication schemes based on the zero-knowledge protocol include zero knowledge proofs of identity by Feige et al [FFS87], efficient zero-knowledge identification scheme for smart cards by Beth [Bet88], zero knowledge authentication scheme with secret key exchange by Brandt et al [BDLP88] and knowledge-proof based versatile smart card verification protocol by Nyang et al [NS00].

We will now state the advantages and disadvantages of zero-knowledge based authentication. The first advantage is that there is no need for a centralised authority which makes the scheme suitable for an ad hoc environment. Another positive feature is that the protocol does not involve passing any kind of secrets between users. Moreover, "an intruder who uses a recorded message can only play back the recorded message if the questions asked happen to be the same" [WYOP94] which makes it less vulnerable to replay attacks.

However, there is one question that must be asked. That is, how does the user B know that the user A can be trusted to initiate and be part of the authentication process in the first place? In other words, from the authentication mechanism in [WYOP94], there is no guarantee that A, who begins the authentication process, is not an attacker. Another problem of the scheme presented in [WYOP94] is the lack of mutual authentication. From [WYOP94], it can be seen that only the party B actually authenticates the party A, not the other way around. The lack of mutual authentication could make the network vulnerable to such attacks as man-in-the-middle attacks.

On the whole, with the zero-knowledge method, authentication can be achieved without revealing any secrets. This protocol also provides some mechanisms against some attacks, such as replay attacks. However, there are still questions to be raised, whether or not the parties involved in the process can be trusted. There is also a shortage of mutual authentication.

3.1.5 Resurrecting duckling

This approach is introduced by Stajano and Anderson in their resurrecting duckling security policy model [SA99]. Stajano extended the idea of the resurrecting duckling further, and presented it in [Sta01]. Both papers focus on wireless ad hoc devices in consumer electrical, medical and industrial appliances.

The resurrecting duckling involves two principals, the duckling (or the slave) and the mother duck (or the master). The mother duck is the one who gives the duckling its soul (or a shared secret that binds the duckling to its mother). "As long as the soul is in the body, the duckling will stay faithful to the mother and obey no one else" [Sta01]. When the association between the duckling and the mother duck ends, the *death* command is issued, and the duckling becomes ready for a new shared secret or a new session. This process is called *reverse metempsychosis*.

In authentication, Stajano and Anderson note the absence of a central server, and suggest that a shared secret is transferred from the mother duck to her duckling by physical contact. This method, the authors claim, eliminates the ambiguity about which two principals are actually involved in the process.

The resurrecting duckling security model is based upon a master-slave relationship. This master-slave bond can only be broken by a master (mother duck) issuing a death command or a timeout to the secret shared (between them). This model is, therefore, secure in this sort (master-slave) of scenario. The protocol is also secure against such attacks as man-in-the-middle attacks due to the close distance between two devices when the secret key is being transferred.

The policy, however, is not suitable for large scale ad hoc networks because of the physical contact needed when a master transfers a secret key to a slave. This requirement of physical contact between communicating parties may be too restrictive to some wireless applications.

3.1.6 Other authentication protocols

We have seen a few authentication protocols and keying mechanisms, which have been employed to make mobile ad hoc networks more secure. In this section, we will discuss another authentication protocol, which is mentioned briefly in [BB03]. This technique is called, by its authors, *location-limited authentication*.

The location-limited authentication is performed between two machines, which are, as the name implies, very close to one another. This technique is very secure, claimed the authors, due to the "physical presence and tamper-detection" [BB03]. Having two nodes very close to one another can reassure both parties that they are authenticating the nodes they think they are authenticating. This protocol is similar to the duckling technique in that when authenticating, the participating parties have to be close to one another. However, there is no master-slave relationship between those devices in location-limited authentication. This location-limited technique is also adopted by [BSSW02].

There are also other protocols, such as Secure Spontaneous Interaction [KZ04], which validates the authentication by humans comparing multimedia streams (ordinary devices such as sounds and LEDs will work). This protocol works as follows. The operators of the two mobile devices press a button on each of the devices at the same time. The machine will then interact with one another (e.g. key exchange etc). At the end, some kind of multimedia stream will be displayed. For example, an LED on each device will flash or the devices will play music. The operators will know that the authentication is done successfully by observing the flashing LEDs, or the sound of a tune chosen by the devices. If the LEDs flash at the same time or the music is in tune, then the authentication is a success.

The authors, Kindberg et al, claim that this method enables the users to detect man-in-the-middle attacks if and when they occur during the authentication. The delay of the multimedia stream will be noticeable to the authenticators if the attacks happen.

First of all, location-limited authentication, similar to the resurrecting duckling model, is secure and suitable for applications within very small areas. However, it is not feasible for a larger setting due to the fact that it is required that the authenticating devices have to be very close to each other.

The second protocol, the secure spontaneous interaction is not suitable nor practical for larger scale mobile ad hoc networks. This is because the operators of the two authenticating devices need to be able to see each other's machine in order for the protocol to complete successfully.

In this section, we have introduced existing authentication protocols and keying mechanisms. We have also given reasons why they fail to satisfy the physical restrictions of pure mobile ad hoc networks. We have seen that a lot of effort has been put in to make mobile ad hoc wireless networks more secure. In the next section, we shall provide an overview of the existing (secure) handoff protocols.

3.2 (Secure) Handoff protocols

A handoff process in a mobile ad hoc network occurs when a mobile node moves beyond the radio range of another mobile node, and enters the coverage of a new node. There are two types of wireless handoff, *hard handoff* and *soft handoff*. A hard handoff is a break-before-make connection. That is, a mobile machine drops the connection with the current node prior to making a new connection with a new node. A soft handoff is essentially a make-before-break connection. That is, a mobile node attempts to make a link to a new node before dropping the connection with the old one.

The movement patterns of a mobile node are classified into two types, *macro-mobility* and *micro-mobility*. Macro-mobility is the movement between different domains, where a domain is a wireless network under a single authority. On the other hand, micro-mobility covers the management of the movement of mobile nodes within a local scope inside a given wireless network [RB01].

We focus our work on micro-mobility in mobile ad hoc networks. The reason for choosing to work on this type of movement is that micro-mobility introduces a problem that does not occur in the macro-mobility management. That is, in micro-mobility there are no extra third party, such as access points and central servers, to help mobile nodes deal with the roaming process. Mobile nodes have to work among themselves only in order to achieve the required handoff. Having extensively looked at the published publications, we claim that no work has been done specifically on (the security of) this type of movement in pure ad hoc networks, and it is only the inter-domain movement of a mobile device that a lot of the work concentrates on. [Rep03] even suggests that WLAN roaming is almost non-existent. Moreover, handoff mechanisms are not mentioned in any of the existing routing protocols. This view is supported by [TGLN05]. Therefore, the work presented in this section will be an attempt to give an overview of what a handoff or roaming process is and how it is handled in wireless infrastructure (rather than infrastructureless) networks.

We will present the following protocols: Mobile IP [Per96], IEEE802.11 MAC layer handoff process [MSA03], various fast handoff schemes [PC02b, PC02a, CP96, TLP99, PC03], the Sabino System [ABABD03] and a protocol explained in [VA00].

After the description of each of the handoff protocols, we will carry out an analysis and explain why they are not suited to mobile ad hoc networks. We, again, should keep in mind when examining the protocols that the problems we have here are very similar to the ones mentioned in the previous section. We have to overcome the physical constraints of mobile ad hoc networks, namely no central authority, bandwidth limitations, computational power limitations, battery life limitations and physical security limitations.

We have mentioned earlier that our work focuses on designing and developing a handoff protocol for pure mobile ad hoc networks only. By *pure*, we mean that there must be no extra pieces of hardware, such as a home/foreign agent, a router and a server, present in the network. In other words, a pure mobile ad hoc network is a network that consists of mobile devices communicating with one another only (no other wired networks or components such as access points are parts of the network or parts of the handoff process).

Therefore, handoffs in mobile ad hoc networks must be carried out without relying on or involving any third party. When a handoff takes place, the only participating parties must be the roaming node itself, the old node (the node that the roaming node is going away from) and the new node (the node whose coverage the roaming node is entering).

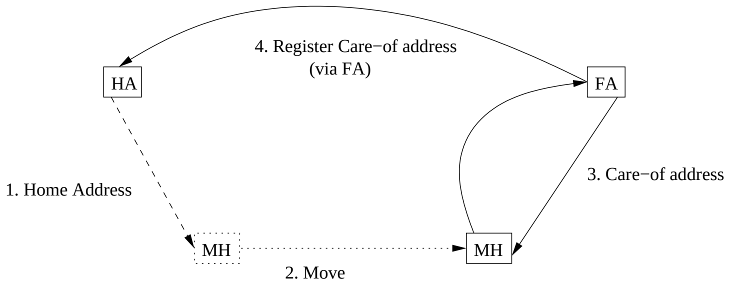
3.2.1 Mobile IP

Mobile IP or IP Mobility Support was proposed by C Perkins and presented in [Per96]. Mobile IP is intended to enable mobile nodes to move from one IP subnet to another. The protocol was designed to handle the problems of IP address of a node and packet forwarding when a node moves from the *home* network to a *foreign* network. The main components of Mobile IP are mobile host (MH), home agent (HA), foreign agent (FA), home address and care-of address. These terms are defined below.

- Mobile host: A mobile node that is allowed to change its point of attachment within a network.
- Home agent: A machine that forwards packets to the mobile host when away from the home network, and maintains current location for the mobile node. The home agent also generates and gives the mobile node its home address.
- Foreign agent: Similar to the home agent, but on the mobile host's visited network. The foreign agent also generates and gives the visiting node its care-off address.
- Home address: The permanent IP address given to a mobile node by the home agent.
- Care-of address: A temporary IP address given to a mobile node by the foreign agent when the mobile node is visiting the network.

A mobile node is given a long-term or permanent IP address by the home agent on a home network. This is called a home address. When the mobile host moves away from the home network to a new foreign network, a new temporary address, called a care-of address, is given to the mobile host by a foreign agent.

The care-of address is associated to the mobile node as long as the node is on that foreign network. The care-of address also acts as a point of location to the home agent, which needs to know the mobile node's care-of address, hence its location. On the foreign network, the mobile host uses its home address as the source address when it sends any IP datagram. The messages that are destined to the mobile node have the node's home address as the destination address. The home agent will intercept the incoming packets and forward them to the mobile node using the care-of address as the destination address. The basic protocol can be summarised in Figure 3-1.



When there is an incoming packet for MH, HA intercepts and forwards it to MH using MH's care−of address.

Figure 3-1: The basic Mobile IP process

Authentication between mobile nodes and home or foreign agent is done using the default keyed MD5 with a 128-bit key. The author also suggests, in [Per96], some techniques that can help to prevent replay attacks. They include the use of timestamps and nonces. [Per96], however, does not explain how data privacy and key management can be handled.

There is a problem with the Mobile IP protocol. That is, during the handoff, packets are often lost, therefore, the home agent may hold out-of-date location information for the mobile node. Furthermore, frequent handoffs mean high overhead and further packet loss. More importantly, as stated in [Per96], Mobile IP is designed to solve the macro-mobility management problem, as a result it is less well suited for a more micro-mobility type movement, which is the problem that we are trying to solve.

3.2.2 Fast inter-AP handoff using predictive authentication scheme in a public WLAN

Pack and Choi developed this fast handoff protocol, which they introduced in [PC02a]. The key idea of this protocol is that a mobile node authenticates with multiple access points rather than with only one access point. An example of protocol applications is depicted and described below.

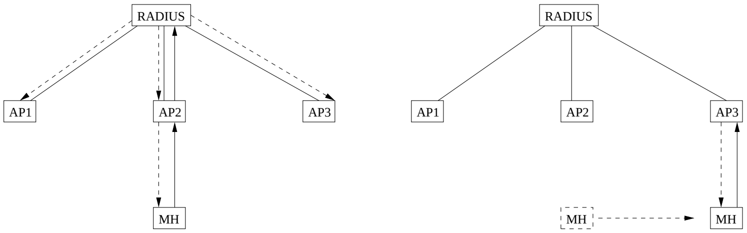


Figure 3-2: The fast inter-AP handoff using predictive authentication

From Figure 3-2, the left figure represents the process when a mobile host first authenticates within the network. The figure on the right shows what happens when the mobile node moves from one access point to another.

Figure 3-2 shows that when a mobile node wishes to join the network, it sends an access request packet to AP2, which forwards the received packet to the RADIUS server [RLMD00]. The RADIUS then sends multiple authentication responses to AP1, AP2 and AP3. The message from the RADIUS to the access points should contain such important data as key information. The access points will remain in, what the authors call, the "soft" state. During the soft state, if there are no handoffs within a specific time period, the key information will be deleted. When the mobile host moves from AP2 to AP3, only the (re)authentication between the mobile node and AP3 occurs. No other message exchanges (e.g. AP3 with RADIUS) are needed.

The other key issue in this protocol is how to select the neighbouring access points to authenticate in advance. It is not efficient to select all of the access points adjacent to the current one. Any movement patterns of a mobile node and the access point's geographical location will have to be taken into account also. The authors introduce an algorithm called the Frequent Handoff Region (FHR) Selection, where FHR is "a set of adjacent APs (which is) determined by factors such as the AP's location in a wireless LAN service area and users' movement pattern" [PC02a]. Informally, the FHR set consists of access points that mobile nodes are likely to move to.

There are two steps to the FHR selection algorithm. The first step is to create a weighted bi-directional graph of access point placement. The weight value is determined by traffic information and mobility pattern. By doing that, we obtain an N by N weight matrix, where N is the number of access points. The next step is to select the frequent handoff region or FHR, using the weight matrix, a weight bound[3] and a specified maximum hop count[4]. The pseudocode and detailed algorithm can be found in [PC02a].

Similar schemes are proposed in [PC02b, PC03]. In [PC02b], the authors also suggest that a mobile node entering a coverage of an access point (or entering a network) should perform authentication with a set of access points, rather than just one access point. This method can minimise the time taken to associate with a new access point when the mobile node moves.

In Prediction-Based Fast Handoff for Mobile WLANs [PC03], the authors propose that their scheme predicts handoff by making use of the moving pattern of mobile nodes. Based on two particular mobility characteristics, the moving pattern tends to be predictable. The first characteristic is that the mobile route is limited. In other words, vehicles such as buses and trains can only follow some predefined paths. The other characteristic is the regularity. An example of this is that public transport tend to run regularly according to their schedules. The authors use these facts to calculate when the next handoff should be executed.

Both [PC02a] and [PC02b] introduce similar algorithms for mobile nodes to select access points to authenticate. These algorithms work based on a couple of facts, one of which is the geographical position of each of the access points within a network. One of the properties of mobile ad hoc networks is their topological instability. Since each mobile node does not have a fixed position, the algorithms proposed in [PC02a] and [PC02b] are not applicable to mobile ad hoc networks.

[3] The value represents the user's service class level. If the weight bound value is high, the user will authenticate with more access points.

[4] This value provides for the possibility of multi-hop handoff.

3.2.3 IEEE802.11 MAC layer handoff process

The IEEE802.11 MAC layer handoff process is described in [MSA03]. Mishra, Shin and Arbaugh (the authors) explain that “The handoff process refers to the mechanism or sequence of messages exchanged by access points and a station resulting in a *transfer* of physical layer connectivity and state information from one AP to another with respect to the station in consideration. Thus the handoff is a physical layer function carried out by at least three participating entities, namely the *station*, a *prior-AP* and a *posterior-AP*” [MSA03]. The handoff process is divided into two phases, the *discovery* phase and the *re-authentication* phase.

The discovery phase is the process that helps a mobile node find a new access point to associate with. When a mobile station moves away from an old access point, the signal degrades, therefore, causes the mobile node to lose connectivity and initiate the handoff. The mobile host can find a new access point by using a MAC layer function, *scan*, which can be categorised into two methods: *active* and *passive*. In the former method, a mobile host sends a broadcast message and listens to responses from the available access points. In the latter method, the mobile station listens to some messages and creates a list in the order of signal strength.

The re-authentication phase involves the transfer of some credentials and state information of the mobile node from the old access point to the new access point. The rest of the re-authentication phase is that two to four messages can be exchanged between the mobile node and the new access point, depending on which authentication method is used.

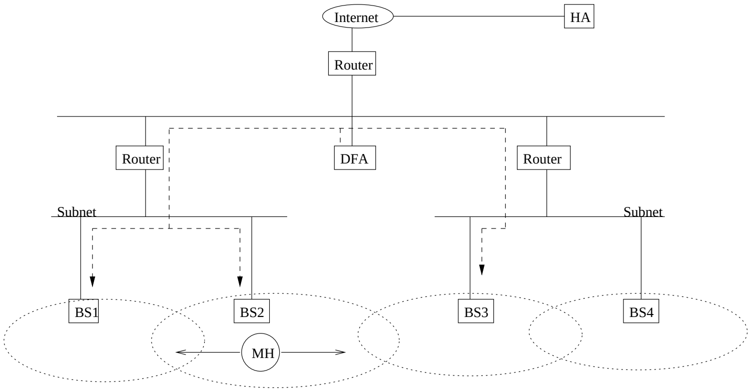
The IEEE802.11 MAC layer handoff protocol does not appear to be suitable for mobile ad hoc networks because of the involvement of access points. The performance of the protocol, as seen in [MSA03] ([MSA03] shows that on average it takes approximately 200 - 400 milliseconds to complete the handoff process), makes it less desirable to be employed in mobile ad hoc networks.

3.2.4 Fast handoff protocols

We will describe two handoff protocols that fall into this category. They are a Fast Handoff Scheme for Wireless Networks [TLP99] and Fast and Scalable Handoffs for Wireless Internetworks [CP96]. There are also other handoff protocols that fall into this category. They include [KP01] and [BHE+04].

The first protocol, a Fast Handoff Scheme for Wireless Networks, is designed by Tan, Lye and Pink, and is documented in [TLP99]. The scheme presented in the paper mainly applies the ideas of Mobile IP (Section 3.2.1). That is it uses the notions of home and foreign agents, and home and care-of addresses.

The major difference is that the authors introduce a hierarchical structure in the foreign network. The main component of the hierarchical structure is the *domain foreign agent* or DFA, which controls various aspects of the foreign network including the care-of address and the forwarding of datagrams. This fast handoff protocol concentrates on the roaming of mobile nodes within one (infrastructure wireless network) domain, namely within the DFA domain. The scheme should become more intuitive if explained with the aid of Figure 3-3. The diagram represents a typical network setup[5].



The dotted arrows represent multicast streams.

Figure 3-3: The hierarchical structure of the protocol

The hierarchical structure approach works as follows. When a mobile host registers with the domain foreign agent, it sends the IP address of the DFA to its home agent as its care-of address. When the mobile host moves from BS_i to BS_j in the same domain, no location update is sent to the home agent. As far as the home agent is concerned, the mobile node is still in the DFA's domain, and

[5] If mobile IP were employed, there would be a foreign agent on each subnet, rather than just one DFA.

all the packets destined for the mobile node are forwarded to the DFA.

When a mobile node roams into a new domain and registers with the DFA, the DFA assigns a multicast address (which is used as the packet forwarding mechanism from the DFA to the base stations (BSs) surrounding the mobile node) unique within its domain. The mobile node informs its serving BS to subscribe to this multicast group. The BS then informs the adjacent BSs to subscribe to the same group. The job of the adjacent BSs is to buffer the recent few packets forwarded to the mobile host by the DFA. This way, as long as the mobile node remains within the same domain, the DFA can be sure that the mobile node will receive packets. The use of the multicast as the packet forwarding mechanism eliminates the generation of location update traffic back to the DFA.

The second protocol, Fast and Scalable Handoffs for Wireless Internetworks [CP96], applies a different hierarchical approach. The authors state that the hierarchical mobility management scheme can handle three cases; local mobility, mobility within an administrative domain and global mobility. The paper [CP96] appears to focus on the local handoff protocol. Figure 3-4 shows the message exchange during a handoff.

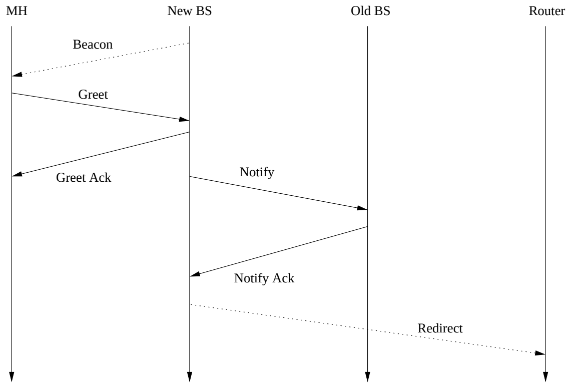


Figure 3-4: Message exchange during a handoff

When a mobile node is in the overlapping area of two base stations or access points, the mobile node sends a `Greet` message to the new base station. The new base station responds with a `Greet Ack` message and creates a routing entry for the mobile node. The new base station then sends a `Notify` packet in order to inform the old base station that the node has moved. After the `Notify Ack` is received, the new base station broadcasts a `Redirect` message to inform any interested nodes and routers that the mobile host has now moved to a new position, and the new base station is now the serving access point. No security protocols are considered in [CP96].

These fast handoff schemes are not applicable to pure mobile ad hoc networks because of the extra third parties, including home and foreign agents, needed to complete the protocol. These agents will not be available in pure mobile ad hoc networks. [CP96], though, does have a desirable feature in that it takes just a few messages to complete the handoff protocol. However, as suggested earlier, the protocol has no security considerations.

3.2.5 Other handoff protocols

There are a couple other handoff protocols that should not be overlooked. They include the Sabino System [ABABD03] and a description of the movement of mobile nodes between clusters in [VA00]. We will briefly describe each of them below.

In Novel Authentication scheme for Ad Hoc Networks (or cluster based authentication) [VA00], the authors briefly mention what needs to be performed when a mobile node moves around the network. Specifically, when a node moves from its current cluster to a new cluster, a mutual authentication is performed between the node and the new cluster head using the system key pair (see Section 3.1.3). The old cluster head removes the entry of the node when it does not receive *Hello* messages for a certain predefined time interval.

The Sabino System involves the exchange of messages between the old and new access points and a roaming mobile node. Those exchanged messages may include such information as the state information of the mobile node and the information needed for identifying the moving node. In the Sabino system, the old access point is the one that makes the decision for the mobile node. In other words, the access point will send a message (which includes the address or location of the destination access point) telling the mobile node to move. The decision is based on the load on the access point. If the load exceeds the threshold, then it is necessary to notify one or more mobile nodes to find a new access point. This approach makes the decision-making process transparent to the mobile node. A more detailed description of the protocol can be found in [ABABD03].

None of the protocols described above are applicable to mobile ad hoc networks, mainly because of the extra hardware required. In order for all of those protocols to work, access points or cluster heads, which are responsible for transferring some of mobile hosts' information, will need to be present as well. This is especially the case for the Sabino system in which access points are the ones which decide when a handoff should occur.

In this section, we have given descriptions of the existing handoff protocols and explained why they are not suited to be employed in mobile ad hoc networks.

Summary

In this chapter, we have introduced the work that is related to our research. We have given a description of the existing authentication and keying mechanisms for mobile ad hoc networks. We have pointed out, after having extensively searched through the published work, that no work has been done specifically on the micro-mobility of handoff protocols in mobile ad hoc networks. Nevertheless, we have attempted to describe the functions of some of the available handoff protocols that have been proposed to be employed in infrastructure wireless networks.

In addition, we have explained that it is more difficult to design and implement authentication and handoff protocols in mobile ad hoc networks due to their physical constraints, which include low bandwidth, low battery life, low computational power, no central authority and lack of physical security. We have analysed and explained why the existing authentication and keying mechanisms are not appropriate for pure ad hoc networks. We have also pointed out that currently there are no handoff protocols that are well suited for mobile ad hoc networks, and why the existing ones are not applicable to the ad hoc environment.

In the next chapter, we will present the designs, analyses and resultant protocols that we believe to be able to solve the problems that the existing protocols have, and to be applicable to pure ad hoc wireless networks.

Chapter 4

Solutions

In this chapter, we design and analyse our new authentication and secure handoff protocols, which can overcome the physical constraints of mobile ad hoc networks and the problems of the existing protocols, stated in the previous chapter, by following the BCK formal model [BCK98a, BCK98b] and applying various theorems and formal definitions [Kra01, CK01]. Furthermore, the correctness of the protocols is then proved using the GNY logic. The reasons for choosing the BCK, CK and GNY methods of analysis are explained in Sections 2.4 and 2.5 respectively. The description of each protocol is provided thereafter. In addition, when designing and describing the protocols, we try to follow the principles described in [AN94] as close as possible.

4.1 Authentication and key establishment

The problem we are trying to solve here is that we would like to prevent unauthorised mobile machines from freely "joining" an existing private local mobile ad hoc network. A way to overcome this problem is authentication. *Authentication* is a process carried out by two parties in order to identify one another. Without authentication, an unauthorised node could "come in" and use the available resources within the network. The problem becomes worse if the unauthorised node is a malicious user. Therefore, it is necessary to have a mechanism for preventing an "outsider" from being part of and compromising the network.

4.1.1 Design

There are three main goals that need to be achieved when designing an authentication protocol for mobile ad hoc networks. They are:

- mutual authentication;
- pairwise session key establishment; and
- efficiency, i.e. having as few messages as possible and the lengths of the messages should be as short as possible.

Mutual authentication is essential, because each node needs to know that the other party is who he says he is, and the other party is authorised to be part of the existing ad hoc network. Note that it is not enough for a joining node to be authenticated by an existing member only. Both nodes must be authenticated by one another.

The establishment of a pairwise session key after mutual authentication is necessary, because we will want to provide privacy for future communication between the two participating parties. Using a shared group key for encryption does not suffice the privacy requirement, as other parties (who also hold the shared secret) could easily decrypt and read the messages. The pairwise session key, established by two communicating nodes, is the way to ensure the privacy.

Finally, the protocol will need to be efficient due to the physical constraints of mobile ad hoc networks that we explained earlier. We will, therefore, want to involve as few messages as possible in the process.

As an assumption (and perhaps a necessity), every mobile node in a domain needs to hold a shared secret, K. By holding the shared secret, each node can prove that he is allowed to be a member of a local private ad hoc network. The second assumption is that all of the public keys have been certified by a certificate authority or CA.

Our authentication protocol consists of two main parts, mutual authentication and key establishment. First, for the mutual authentication purpose, we will use the challenge-and-response technique. This part of the protocol allows each party to assure the other party that he is also holding the same secret key, K, and hence is allowed to be part of the network. The second half of the protocol is the pairwise session key establishment. We will apply the Diffie-Hellman method to achieve this.

Furthermore, if a node is not already a member of the existing network, all the packets he sends will be quietly discarded by the existing members, except for the *request-to-join* message. This mechanism is designed to prevent a malicious mobile node from injecting forged messages into the network.

The next section contains the construction, analyses and proofs of an authentication and key establishment protocol.

4.1.2 Analysis and construction

We will construct, analyse and prove an authentication protocol by applying the BCK modular method [BCK98a, BCK98b] as well as the CK analysis method [CK01]. The correctness of the protocol will be analysed and proved using the GNY analysis [GNY90]. The complete description of the resultant protocol will be presented at the end of this section.

Before we begin, we would like to introduce the theorems and formal definitions that are essential for the analyses and proofs of the protocol.

- **Theorem (1 of [Kra01])**: *If ENC is a symmetric encryption scheme secure in the sense of IND-CPA*[1] *and MAC is a secure MAC family (that takes ciphertext and a secret key as its inputs) then method $EtA(ENC, MAC)$ implements secure channels (where EtA means encrypt-then-authenticate scheme).*

  The above theorem means that if a network packet is composed in such a way as $EtA(ENC, MAC)$, then the secrecy and authenticity of that particular network channel is guaranteed. This is under the assumptions that then MAC scheme is secure and the encryption function is secure in the following sense. If an attacker sends a packet to an honest user containing two messages, m_1 and m_2, and he receives a reply, $C = E(m_b)$, the attacker must not be able to guess correctly the value of b with a probability significantly greater than 1/2.

[1] IND-CPA means indistinguishability of encryptions against chosen plaintext attack.

- **Definition (2 of [BCK98a, BCK98b])**: *A* `compiler` $\mathcal{C}$ *is an algorithm that takes for input descriptions of protocols and outputs descriptions of protocols. An* `authenticator` *is a compiler* $\mathcal{C}$ *where for any protocol* π*, the protocol* $\mathcal{C}(\pi)$ *emulates* π *in unauthenticated networks.*

  The more detailed definitions of a *compiler* and an *authenticator* can be found in Section 2.4.

- **Theorem (8 of [CK01])**: *Assuming the Decisional Diffie-Hellman (DDH) assumption, protocol Diffie-Hellman is SK-secure in the authenticated-links model.*

  The DDH assumption is stated in [CK01] as follows. Let k be a security parameter. Let p, q be primes, where q is the length k bits and $q/p-1$ and g be of order q in Z_p^*. Then the probability distributions of quintuples $Q_0 = \{< p, g, g^x, g^y, g^{xy} >: x, y \overset{R}{\leftarrow} Z_q\}$ and $Q_1 = \{< p, g, g^x, g^y, g^z >: x, y, z \overset{R}{\leftarrow} Z_q\}$ are computationally indistinguishable. By *computationally indistinguishable*, we mean that given Q_0 and Q_1, one cannot decide which is which with a probability of success non-negligibly better than random guessing.

- Two other essential definitions are the definition of *SK-security* (Definition 4 of [CK01]) and the definition of *secure network channels* (Definition 15 of [CK01]). Both of these are explained in detail in Section 2.4.

We split the authentication and key exchange protocol into three parts or modules for the analysis. We show that each of the three parts is secure in the unauthenticated-links model. We use the following notations throughout this section.

- E_{+K_i} - an asymmetric encryption function with i's public key
- D_{-K_i} - an asymmetric decryption function with i's private key
- $E_K()$ - a symmetric encryption function with key K
- $D_K()$ - a symmetric decryption function with key K
- $MAC()$ - a message authentication code function [KBC97]
- N_i - a random nonce generated by i

Part 1 - Advertising public key components

In this part, the joining mobile node will advertise his or her public key to the existing members of the network.

Step 0: Initialisation: A computes $+K_a, -K_a$. B computes $+K_b, -K_b$. And (as an assumption) they both have a shared secret, K.

$+K_i$: public key of i,

$-K_i$: private key of i

Step 1: The initiator (the node that wishes to join the network) B chooses a sequence number $seq \overset{R}{\longleftarrow} \{0,1\}^n$ and a random nonce $N_b \overset{R}{\longleftarrow} \{0,1\}^n$, where n is 32 bits. B computes $C = E_K(+K_b, N_b, seq)$ and $MAC(C, K)$, and sends $(Req, C, MAC(C, K))$ to A, an existing member.

Step 2a: Upon receipt of $(Req, C, MAC(C, K))$, A checks the message integrity and computes $m = D_K(C)$. If both are successful then A holds N_b and the sequence number, seq.

Lemma 4.1.1. *Assume that $E_K()$ is secure against chosen-plaintext attacks*[2] *and the MAC scheme used is secure. Then we claim that the above steps constitute a secure channel.*

Proof. It can be seen that within this part of the protocol, only one message is sent. By Theorem 1 in [Kra01], we can see that the way the packet is constructed, i.e. encrypt-then-authenticate, implements a secure channel. Hence the secrecy and authenticity is guaranteed.

By Definition 15 (*a secure network channel*) in [CK01], we can conclude that if the channel is secure then a secure network encryption protocol and also a secure network authentication protocol are achieved. This, in addition, implies that the network channel, and therefore this part of the protocol, is secure in the unauthenticated-links model. □

Part 2 - Mutual authentication

(Carrying on from Step 2a in Part 1)

This part uses the challenge-and-response technique for mutual authentication between the joining node and the existing member.

[2] A chosen-plaintext attack is one where the adversary chooses plaintext and is then given corresponding ciphertext. Subsequently, the adversary uses any information deduced in order to recover plaintext corresponding to previously unseen ciphertext [MvOV96].

Step 2b: A chooses a challenge $R_A \xleftarrow{R} \{0,1\}^d$ and a random nonce $N_a \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits. A increments seq by 1. Note that A "sees" seq, N_b and $+K_b$ from computing $m = D_K(C)$. A sends $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$ to B. A computes $E_K(R_A)$, which will be needed for the verification of the challenge.

Step 3: Upon receipt of $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$. B decrypts the message $m = D_{-K_b}(E_{+K_b}(N_a, N_b, seq, R_A, +K_a))$. B computes $E_K(R_A)$.

Before we begin the proof, we have to state that the above protocol (Part 2) can be thought of as a simple key exchange protocol. That is P_1 chooses some value, sends it to P_2. They then use the same (pseudorandom) function, with that value as the input, to calculate the key. In this case, A chooses and sends R_A to B, and computes $E_K(R_A)$. B, after receiving the packet, also computes $E_K(R_A)$. For the analysis and proof purposes, we call this protocol `Temp` which can be formally written as follows.

Protocol Temp:

Note that $+K_i$ is the public key of i, $-K_i$ is the private key of i, and K and $E_K()$ are the key and encryption function known to both parties, respectively.

Step 1: The initiator A chooses $R_A \xleftarrow{R} \{0,1\}^d$ and sends $E_{+K_b}(R_A)$ to B. A computes $E_K(R_A)$. For the sake of argument, we think of $E_K(R_A)$ as the new session "key".

Step 2: Upon receipt of $E_{+K_b}(R_A)$, the responder B computes $m = D_{-K_b}(E_{+K_b}(R_A))$. If the decryption is successful then B computes $E_K(R_A)$, which can also be thought of as the new session "key".

Lemma 4.1.2. *Assume that the public key encryption is secure against chosen-ciphertext attacks*[3] *and* $E_K()$ *is a pseudorandom function*[4]*, then the protocol (steps 2 and 3 of Part 2) is secure in the authenticated-links model.*

[3] A chosen-ciphertext attack is one where the adversary selects the ciphertext and is given the corresponding plaintext. The objective is to be able to deduce the plaintext from (different) ciphertext [MvOV96].

[4] A pseudorandom function is an efficient algorithm that when given an n-bit s and an n-bit x, returns an n-bit $f_s(x)$ such that it is infeasible to distinguish $f_s(x)$ from a truly random function.

Proof. Our proof for this part of the protocol is largely inspired by the proof of Theorem 9 in [CK01].

We need to show that this protocol, `Temp`, is SK-secure [CK01]. According to Definition 4 (SK-secure) in [CK01], it can be seen that the protocol satisfies the first condition. If both A and B are uncorrupted during steps 2 and 3 (or steps 1 and 2 of `Temp`), and both parties complete the exchange, then they both hold R_A or $E_K(R_A)$.

Now it is left for us to prove that the second condition of the SK-secure definition also holds. We start by defining a "game", which captures the chosen ciphertext security of the public key encryption. We will then show that an attacker that breaks the SK-security of our protocol `Temp` can win this game, and break the public key cryptosystem. The game, we call `Game`, is defined below.

`Game` involves two parties $\mathcal{G}$ and $\mathcal{B}$. $\mathcal{G}$ holds both public and private keys, $+K$ and $-K$. $\mathcal{B}$ holds only the public key $+K$. Both parties know the shared secret K and the symmetric encryption function, $E_K()$.

Phase 0: $\mathcal{G}$ provides $\mathcal{B}$ with a challenge ciphertext $c^* = E_{+K}(K_0)$ for $K_0 \xleftarrow{R} \{0,1\}^n$.
Phase 1: $\mathcal{B}$ sends a test ciphertext c to $\mathcal{G}$, who responds with K' where $K' = E_{-K}(c)$. This is repeated a number of times with c being chosen adaptively by $\mathcal{B}$ after receiving $\mathcal{G}$'s response.
Phase 2: $\mathcal{G}$ chooses a random bit, $b \xleftarrow{R} \{0,1\}$. If $b = 0$ then $\mathcal{G}$ sends to $\mathcal{B}$ $E_K(K_0)$. If $b = 1$ then $\mathcal{G}$ sends a random value r of the same length as $E_K(K_0)$. Note that K_0 is the value encrypted by $\mathcal{G}$ in Phase 0.
Phase 3: $\mathcal{B}$ outputs a bit b'. If $b' = b$ then $\mathcal{B}$ wins. In other words, if $\mathcal{B}$ guesses correctly that the value received is real or random, then he wins the `Game`.

By Lemma 4.1.3 shown below, we have shown that the protocol `Temp` satisfies the second condition of Definition 4 in [CK01], which means that the protocol is SK-secure in authenticated-links model. We prove Lemma 4.1.3 below. □

Lemma 4.1.3. *Assume that the public key cryptosystem is secure against chosen ciphertext attacks. Then, with reference to* `Game` *defined above, if c^* is not queried by $\mathcal{B}$, the probability that $\mathcal{B}$ wins is no more than $1/2$ plus a negligible fraction.*

Proof. Assume to the contrary that there is an AM (authenticated-links model) attacker $\mathcal{A}$ that breaks the SK-security of the protocol $\mathtt{Temp}$, in the sense that it can distinguish between real and random values of a test session while not being allowed to corrupt the parties to this session. Then there is an efficient algorithm that wins the $\mathtt{Game}$ with non-negligible probability over 1/2.

The algorithm works as follows. Let $\mathcal{G}$ be the party against which $\mathcal{B}$ plays the $\mathtt{Game}$. $\mathcal{G}$ holds a private key and a public key, $-K$ and $+K$ respectively. The $\mathtt{Game}$ starts with $\mathcal{G}$ sending a challenge ciphertext c^* to $\mathcal{B}$. $\mathcal{B}$ then proceeds to Phase 1 of the $\mathtt{Game}$ doing the following.

$\mathcal{B}$ builds a virtual scenario for the run of the protocol $\mathtt{Temp}$, and activates the attacker $\mathcal{A}$ against this virtual run. Among all n parties in this virtual run, $\mathcal{B}$ chooses one at random, call it P_j^*. For all other parties, $\mathcal{B}$ computes private keys and provides $\mathcal{A}$ with the corresponding public keys. $\mathcal{B}$ does not choose a private key for P_j^*. Instead he provides $\mathcal{A}$ with $+K$ (the public key of $\mathcal{G}$) as the public key of P_j^*. Also, $\mathcal{B}$ chooses a random session among the sessions where P_j^* is the responder and P_i^* is the initiator. We call this session s^*.

All operations scheduled by the attacker $\mathcal{A}$ are performed by $\mathcal{B}$ on behalf of the virtual players in the following way. All session establishments are executed by $\mathcal{B}$ according to the protocol $\mathtt{Temp}$, except for the establishment of the session s^*. When $\mathcal{A}$ schedules to establish the session s^* between a party P_i^* and the chosen responder P_j^*, $\mathcal{B}$ sends c^* on behalf of P_i^* to P_j^*. c^* is the challenge ciphertext provided to $\mathcal{B}$ by $\mathcal{G}$ in Phase 0.

All exposure of session keys performed by the attacker $\mathcal{A}$, via session or party corruptions, that do not involve P_j^* as the responder are answered by $\mathcal{B}$ using his knowledge of private keys. When $\mathcal{A}$ corrupts a party other than P_j^* and P_i^*, $\mathcal{B}$ provides $\mathcal{A}$ with the private key of that party. If a session $s \neq s^*$ between a party P and P_j^*, where P_j^* is a responder, is exposed by $\mathcal{A}$, then $\mathcal{B}$ provides the value of the key in the following way. If P is corrupted at the same time as session s^* is established then $\mathcal{B}$ is the one to choose the key. If P is uncorrupted then all $\mathcal{B}$ knows is the encrypted value, c, sent in Step 1 of the protocol $\mathtt{Temp}$. In this case, $\mathcal{B}$ presents $\mathcal{G}$ (as part of Phase 1) the test ciphertext c. The value returned by $\mathcal{G}$ is the value that $\mathcal{B}$ provides to $\mathcal{A}$ as the queried session key.

If at any point the attacker $\mathcal{A}$ queries or reveals session s^*, corrupts P_i^* or P_j^*, or chooses a test session other than s^*, the run of the attacker $\mathcal{A}$ is aborted and

Phase 2 of the `Game` is executed. After receiving a response from $\mathcal{G}$, $\mathcal{B}$ outputs a bit b'.

If $\mathcal{A}$ chooses the test session s^*, $\mathcal{G}$ sends his Phase 2 response to $\mathcal{B}$. $\mathcal{B}$ then passes that value to $\mathcal{A}$ as the value of the key for session s^*. When $\mathcal{A}$ outputs the bit b', $\mathcal{B}$ also outputs the same bit b'.

We now have to show that $\mathcal{B}$ can win the `Game` against $\mathcal{G}$ with non-negligible probability over 1/2. First, when the run of the attacker $\mathcal{A}$ is aborted, $\mathcal{B}$ outputs a bit b', so his chances of winning in this case is exactly 1/2. In the case where $\mathcal{A}$ outputs b', the chances that $\mathcal{B}$ will win is exactly the same as those of $\mathcal{A}$ to guess correctly whether the value ($\mathcal{G}$'s response) is real or random. This probability is, by our contradiction statement, $1/2 + \varepsilon$ for non-negligible ε. This case here happens when the test session s^* chosen by $\mathcal{A}$ is the same s^* chosen by $\mathcal{B}$. Since this event happens with the probability $1/l$, where l is an upper bound on the number of sessions established in the protocol runs, the overall advantage of $\mathcal{B}$ is non-negligible. □

We have proved that protocol `TEMP` is SK-secure in authenticated-links model.We will now provide a protocol that is secure in the unauthenticated-links model.

In order to make the protocol SK-secure in unauthenticated networks, we will apply the signature-based MT-authenticator, given in [BCK98a, BCK98b], to the steps 2 and 3 above. By Definition 2 in [BCK98a, BCK98b], we can see that the resultant protocol $\pi' = C_{\lambda_{sig}}(\pi)$ is secure in unauthenticated networks.

Below is the protocol we get after applying the signature-based MT-authenticator, λ_{sig}. Formally, let π be the "original" protocol that is secure in the authenticated-links model. Let π' be the resultant protocol that is secure in unauthenticated networks. We have $\pi' = C_{\lambda_{sig}}(\pi)$:

Step 2: A chooses $R_A \xleftarrow{R} \{0,1\}^d$ and $N_a \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits. A increments *seq* by 1. Note that A "sees" *seq*, N_b and $+K_b$ from computing $m = D_K(C)$. A sends $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$ together with his signature $SIG_A(N_a, N_b, seq, R_A, +K_a, B)$. A computes $E_K(R_A)$.
Step 3: Upon receipt of $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$ and A's signature, B computes $m = D_{-K_b}(E_{+K_b}(N_a, N_b, seq, R_A, +K_a))$ and verifies the signature. B only accepts m if the verification of the signature is successful. (We will talk about

the nonce N_b and *seq* later.) B computes $E_K(R_A)$.

Again, by Definition 2 in [BCK98a, BCK98b], we claim that the protocol above is secure in the unauthenticated-links model. But, does the above protocol conform to the signature-based MT-authenticator, λ_{sig}? Here, we can say that we use the nonce N_b, sent to A by B in Step 1 (of Part 1), as the challenge required by the authenticator. A then sends a message to B together with his signature. A also includes the challenge N_b and B's identity in his signature. Note that the identity of B is necessarily included in the signature, otherwise, as explained in [DvOW92], the scheme becomes insecure.

We have shown that this part of the authentication and key exchange protocol is secure in the unauthenticated-links model because of the application of the signature-based MT-authenticator, λ_{sig}.

Part 3 - Pairwise key establishment

The last stage of the authentication and key establishment protocol is the establishment of a pairwise session key that will be used by A and B for the communication between themselves.

We have decided to use a simple two-pass Diffie-Hellman protocol to establish the session-key. The Diffie-Hellman protocol is proved to be SK-secure in the authenticated-links model in [CK01], according to Theorem 8.

In order to make the protocol, hence this stage, secure in unauthenticated networks, we apply the same signature-based MT-authenticator as the previous stage, namely λ_{sig}, to the protocol. We get $\pi' = C_{\lambda_{sig}}(\pi)$ that is:

Step 3: Upon receipt of $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$, B computes $m = D_{-K_b}(E_{+K_b}(N_a, N_b, seq, R_A, +K_a))$ and verifies the signature of A. If the signature is valid then B computes $C = E_K(R_A)$ and increment *seq* by 1. B sends the message $E_{+K_a}(N_a, seq, C, \beta = g^y)$ together with his signature $SIG_B(N_a, seq, C, \beta = g^y, A)$ to A. (We will discuss the nonce and sequence number verification later.) Also note that β is the DH public component of B.

Step 4: Upon receipt of $E_{+K_a}(N_a, seq, C, \beta)$ and B's signature, A computes $m = D_{-K_a}(E_{+K_a}(N_a, seq, C, \beta))$ and verifies the signature. If successful, A accepts the message m (we will again discuss the nonce, sequence number and C

verification later), increments seq by 1 and sends to B $E_{+K_b}(N_b, seq, \alpha = g^x)$ together with his signature $SIG_A(N_a, seq, \alpha = g^x, \beta, B)$. A is now able to compute the pairwise session key $K_{ab} = \beta^x$. Note that $\alpha = g^x$ is the DH public component of A.

Step 5: Upon receipt of $E_{+K_b}(N_b, seq, \alpha)$ and A's signature, B computes $m = D_{-K_b}(E_{+K_b}(N_b, seq, \alpha))$ and verifies the signature. (Again we will talk about the nonce and sequence number verification later.) If the signature is valid then B computes the session key $K_{ab} = \alpha^y$.

Note that we have left the verification of the nonce, sequence number and encrypted challenge for later discussion, because, at this time, we just want to show what the protocol "looks like" after applying the signature-based MT-authenticator, rather than the whole working protocol.

Lemma 4.1.4. *Stage 3 (the above protocol) is secure in the unauthenticated-links model.*

Proof. To show that stage 3 is secure in the unauthenticated-links model, we need to illustrate that the protocol does indeed conform to the signature-based MT-authenticator, λ_{sig}. We look at each step in turn.

In Step 3, we use N_a, the nonce sent to B by A, as the challenge for B to sign. This is adequate, because the nonce N_a is chosen by A both freshly and at random.

In Step 4, we can see that we need to include $\beta = g^y$ (B's DH public component) in the signature. In other words, we use β as a challenge for A to sign. This is necessary because there is nothing fresh (except for β) in the message received by A in this step that can be used as a challenge. The sequence number is fresh, but we do not want to use it as the challenge, because it is incremented by each party before a message is sent.

Then A and B only accept the message if the verification of the other party's signature is successful.

It now appears that our stage 3 behaves as required by the signature-based MT-authenticator, therefore, we can claim, by Definition 2 in [BCK98a], that the exchange of the messages in this stage is secure in unauthenticated networks. □

We now have the three stages, constituting an authentication and key exchange protocol, that are secure in the unauthenticated-links model. Therefore, we conclude that the whole (resultant) protocol is also secure in unauthenticated networks.

4.1.3 Proof of correctness

We will analyse the authentication protocol and pairwise session key establishment by using the GNY logic. The notations and logical postulates of the GNY protocol that are applied here can be found in Appendix A. The authentication protocol written in GNY logic is as follows:

1. B→A: A: $\triangleleft *Req, *\{* + K_b, *N_b, *seq\}_K, *MAC(*\{* + K_b, *N_b, *seq\}_K)$

2. A→B: B: $\triangleleft *\{*N_a, N_b, *R_A, *seq, * + K_a\}_{+K_b}, *\{*H(*N_a, N_b, *R_A, *seq,$ $* +K_a, B)\}_{-K_a}$

3. B→A: A: $\triangleleft *\{N_a, *seq, *\{R_A\}_K, *K''\}_{+K_a}, *\{*H(N_a, *seq, *\{R_A\}_K, *K'', A)\}_{-K_b}$ $\leadsto B \mid\equiv B \overset{K''}{\longleftrightarrow} A$, where $K'' = g^y \bmod p$

4. A→B: B: $\triangleleft *\{N_b, *seq, *K'\}_{+K_b}, *\{*H(N_b, *seq, *K', K'', B)\}_{-K_a}$ $\leadsto A \mid\equiv A \overset{K'}{\longleftrightarrow} B$, where $K' = g^x \bmod p$

Assumptions

$A\mid\equiv A\overset{K}{\longleftrightarrow}B$	$B\mid\equiv B\overset{K}{\longleftrightarrow}A$	$A\mid\equiv A\overset{K'}{\longleftrightarrow}B$
$B\mid\equiv B\overset{K''}{\longleftrightarrow}A$	$A\mid\equiv B\mid\Longrightarrow B\overset{K''}{\longleftrightarrow}A$	$B\mid\equiv A\mid\Longrightarrow A\overset{K'}{\longleftrightarrow}B$
$A\mid\equiv B\mid\Longrightarrow B\mid\equiv *$	$B\mid\equiv A\mid\Longrightarrow A\mid\equiv *$	$A\mid\equiv \sharp(N_a)$
$B\mid\equiv \sharp(N_b)$	$A\mid\equiv \sharp(R_A)$	$A\ni K$
$B\ni K$	$A\ni +K_a$	$B\ni +K_b$
$A\ni -K_a$	$B\ni -K_b$	$A\ni R_A$
$A\ni N_a$	$B\ni N_b$	

Notations:

- $+K_a$ - A's public key
- $-K_a$ - A's private key
- K' - A's DH public component
- $H()$ - a one-way hash function
- $+K_b$ - B's public key
- $-K_b$ - B's private key
- K'' - B's DH public component

Analysis

Message 1:

$$\text{T1: }\ \frac{A \triangleleft *Req, *\{* + K_b, *N_b, *seq\}_K, *MAC(*\{* + K_b, *N_b, *seq\}_K)}{A \triangleleft Req, \{+K_b, N_b, seq\}_K, MAC(\{+K_b, N_b, seq\}_K)}$$

$$\text{T3: }\ \frac{A \triangleleft Req, \{+K_b, N_b, seq\}_K, MAC(\{+K_b, N_b, seq\}_K), A \ni K}{A \triangleleft Req, +K_b, N_b, seq, MAC(\{+K_b, N_b, seq\}_K)}$$

$$\text{P1: }\ \frac{A \triangleleft Req, +K_b, N_b, seq, MAC(\{+K_b, N_b, seq\}_K)}{A \ni Req, +K_b, N_b, seq, MAC(\{+K_b, N_b, seq\}_K)}$$

Since $A \ni +K_b, N_b, seq$ and $A \ni K$, by P4, $A \ni H(\{+K_b, N_b, seq\}_K)$. Now A can check if $H(\{+K_b, N_b, seq\}_K)$ is equal to $MAC(\{+K_b, N_b, seq\}_K)$. If so, carry on.

Message 2:

$$\text{T1: }\ \frac{B \triangleleft *\{*N_a, N_b, *R_A, *seq, * + K_a\}_{+K_b}, *\{*H(*N_a, N_b, *R_A, *seq, * + K_a, B)\}_{-K_a}}{B \triangleleft \{N_a, N_b, R_A, seq, +K_a\}_{+K_b}, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a}}$$

$$\text{T4: }\ \frac{B \triangleleft \{N_a, N_b, R_A, seq, +K_a\}_{+K_b}, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a}, B \ni -K_b}{B \triangleleft N_a, N_b, R_A, seq, +K_a, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a}}$$

$$\text{P1: }\ \frac{B \triangleleft N_a, N_b, R_A, seq, +K_a, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a}}{B \ni N_a, N_b, R_A, seq, +K_a, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a}}$$

$$\text{P4: }\ \frac{B \ni (N_a, N_b, R_A, seq, +K_a), B \ni B}{B \ni H(N_a, N_b, R_A, seq, +K_a, B)}$$

Note that $B \ni B$ because B possesses his own identity anyway.

$$\text{F1: }\ \frac{B \mid\equiv \sharp seq, \sharp N_a}{B \mid\equiv \sharp(N_a, N_b, R_A, seq, +K_a, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a})}$$

$$\text{R1: }\ \frac{B \mid\equiv \phi N_b}{B \mid\equiv \phi(N_a, N_b, R_A, seq, +K_a, \{H(N_a, N_b, R_A, seq, +K_a, B)\}_{-K_a})}$$

Let $(*N_a, N_b, *R_A, *seq, * + K_a)$ be X.

$$\text{R5: }\ \frac{B \mid\equiv \phi(X), B \ni (X, B), B \mid\equiv \phi(B)}{B \mid\equiv \phi(H(X)), B \mid\equiv \phi(H(X, B))}$$

$$\text{I4: } \frac{B \lhd \{H(X,B)\}_{-K_a}, B \ni +K_a, B \mid\equiv \overset{+K_a}{\mapsto} A, B \mid\equiv \phi(H(X,B))}{B \mid\equiv A \mid\sim H(X,B), B \mid\equiv A \mid\sim \{H(X,B)\}_{-K_a}}$$

B believes that A signed the message, and therefore, sent the message. That is $B \mid\equiv A \mid\sim X, H(X,B)$.

Message 3:
$B \mid\equiv B \overset{K''}{\longleftrightarrow} A$ is valid because it is an initial assumption.

$$\text{T1: } \frac{A \lhd *\{N_a, *seq, *\{R_A\}_K, *K''\}_{+K_a}, *\{*H(N_a, *seq, *\{R_A\}_K, *K'', A)\}_{-K_b}}{A \lhd \{N_a, seq, \{R_A\}_K, K''\}_{+K_a}, \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b}}$$

$$\text{T4: } \frac{A \lhd \{N_a, seq, \{R_A\}_K, K''\}_{+K_a}, \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b}, A \ni -K_a}{A \lhd N_a, seq, \{R_A\}_K, K'', \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b}}$$

$$\text{P1: } \frac{A \lhd N_a, seq, \{R_A\}_K, K'', \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b}}{A \ni N_a, seq, \{R_A\}_K, K'', \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b}}$$

A can now verifies $\{R_A\}_K$.

$$\text{P4: } \frac{A \ni (N_a, seq, \{R_A\}_K, K''), A \ni A}{A \ni H(N_a, seq, \{R_A\}_K, K'', A)}$$

Note that $A \ni A$ because A possesses his own identity anyway.

$$\text{F1: } \frac{A \mid\equiv \sharp seq}{A \mid\equiv \sharp(N_a, seq, \{R_A\}_K, K'', \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b})}$$

$$\text{R1: } \frac{A \mid\equiv \phi N_a}{A \mid\equiv \phi(N_a, seq, \{R_A\}_K, K'', \{H(N_a, seq, \{R_A\}_K, K'', A)\}_{-K_b})}$$

Let $(N_a, *seq, *\{R_A\}_K, *K'')$ be X.

$$\text{R5: } \frac{A \mid\equiv \phi(X), A \ni (X, A), A \mid\equiv \phi(A)}{A \mid\equiv \phi(H(X)), A \mid\equiv (H(X,A))}$$

$$\text{I4: } \frac{A \lhd \{H(X,A)\}_{-K_b}, A \ni +K_b, A \mid\equiv \overset{+K_b}{\mapsto} B, A \mid\equiv \phi(H(X,A))}{A \mid\equiv B \mid\sim H(X,A), A \mid\equiv B \mid\sim \{H(X,A)\}_{-K_b}}$$

A believes that B signed the message, and therefore, sent the message. That is $A \mid\equiv B \mid\sim X, H(X, A)$.

$$\text{J2: } \frac{A \mid\equiv B \mid\Longrightarrow B \mid\equiv *, A \mid\equiv B \mid\sim (X \leadsto B \mid\equiv B \xleftrightarrow{K''} A), A \mid\equiv \sharp(X)}{A \mid\equiv B \mid\equiv B \xleftrightarrow{K''} A}$$

$$\text{J1: } \frac{A \mid\equiv B \mid\Longrightarrow B \xleftrightarrow{K''} A, A \mid\equiv B \mid\equiv B \xleftrightarrow{K''} A}{A \mid\equiv B \xleftrightarrow{K''} A}$$

which now implies that $A \mid\equiv B \xleftrightarrow{K_{ab}} A$. After this message, A believes that the newly computed key, K_{ab}, is shared between A and B.

Message 4

$A \mid\equiv A \xleftrightarrow{K'} B$ is valid because it is an initial assumption.

$$\text{T1: } \frac{B \lhd *\{N_b, *seq, *K'\}_{+K_b}, *\{*H(N_b, *seq, *K', K'', B)\}_{-K_a}}{B \lhd \{N_b, seq, K'\}_{+K_b}, \{H(N_b, seq, K', K'', B)\}_{-K_a}}$$

$$\text{T4: } \frac{B \lhd \{N_b, seq, K'\}_{+K_b}, \{H(N_b, seq, K', K'', B)\}_{-K_a}, B \ni -K_b}{B \lhd N_b, seq, K', \{H(N_b, seq, K', K'', B)\}_{-K_a}}$$

$$\text{P1: } \frac{B \lhd N_b, seq, K', \{H(N_b, seq, Ki, K'', B)\}_{-K_a}}{B \ni N_b, seq, K', \{H(N_b, seq, K', K'', B)\}_{-K_a}}$$

$$\text{P4: } \frac{B \ni (N_b, seq, K'), B \ni (K'', B)}{B \ni H(N_b, seq, K', K'', B)}$$

Note that $B \ni B$ because B possesses his own identity, and $B \ni K''$ because K'' is B's Diffie-Hellman public component.

$$\text{F1: } \frac{B \mid\equiv \sharp seq}{B \mid\equiv \sharp(N_b, seq, K', \{H(N_b, seq, K', K'', B)\}_{-K_a})}$$

$$\text{R1: } \frac{B \mid\equiv \phi N_b}{B \mid\equiv \phi(N_b, seq, K', \{H(N_b, seq, K', K'', B)\}_{-K_a})}$$

Let $(N_b, *seq, *K')$ be X.

$$\text{R5: } \frac{B \mid\equiv \phi(X), B \ni (X, B), B \mid\equiv \phi(B)}{B \mid\equiv \phi(H(X)), B \mid\equiv \phi(H(X, B))}$$

I4: $$\frac{B \lhd \{H(X,B)\}_{-K_a}, B \ni +K_a, B \mid\equiv \overset{+K_a}{\mapsto} A, B \mid\equiv \phi(H(X,B))}{B \mid\equiv A \mid\sim H(X,B), B \mid\equiv A \mid\sim \{H(X,B)\}_{-K_a}}$$

B believes that A signed the message, and therefore, sent the message. That is $B \mid\equiv A \mid\sim X, H(X,B)$.

J2: $$\frac{B \mid\equiv A \mid\Longrightarrow A \mid\equiv *, B \mid\equiv A \mid\sim (X \rightsquigarrow A \mid\equiv A \xleftrightarrow{K'} B), B \mid\equiv \sharp(X)}{B \mid\equiv A \mid\equiv A \xleftrightarrow{K'} B}$$

J1: $$\frac{B \mid\equiv A \mid\Longrightarrow A \xleftrightarrow{K'} B, B \mid\equiv A \mid\equiv A \xleftrightarrow{K'} B}{B \mid\equiv A \xleftrightarrow{K'} B}$$

which now implies that $B \mid\equiv A \xleftrightarrow{K_{ab}} B$. At this stage, B now believes that the newly calculated key, K_{ab}, is shared between the parties A and B.

Therefore, we get:

$$\begin{array}{lll} A \mid\equiv B \xleftrightarrow{K''} A & \text{and} & B \mid\equiv A \xleftrightarrow{K'} B \\ A \mid\equiv B \mid\equiv B \xleftrightarrow{K''} A & & B \mid\equiv A \mid\equiv A \xleftrightarrow{K'} B \end{array}$$

which imply that

$$\begin{array}{lll} A \mid\equiv B \xleftrightarrow{K_{ab}} A & \text{and} & B \mid\equiv A \xleftrightarrow{K_{ab}} B \\ A \mid\equiv B \mid\equiv B \xleftrightarrow{K_{ab}} A & & B \mid\equiv A \mid\equiv A \xleftrightarrow{K_{ab}} B \end{array}$$

where $K_{ab} = (g^x)^y \bmod p = (g^y)^x \bmod p$.

At the end of the authentication protocol, we have achieved the following outcomes. The parties A and B believe that the key, K_{ab}, is shared between themselves. Each of them also believes that the other party believes that the key, K_{ab}, is a suitable secret for A and B.

4.1.4 Description of the resultant protocol

We will now summarise the authentication and key establishment protocol that we believe to be secure in the unauthenticated-links model. Again, the following notations are used throughout the description of the protocol.

- E_{+K_i} - an asymmetric encryption function with i's public key
- D_{-K_i} - an asymmetric decryption function with i's private key

- $E_K()$ - a symmetric encryption function with key K
- $D_K()$ - a symmetric decryption function with key K
- $MAC()$ - a message authentication code function [KBC97]
- N_i - a random nonce generated by i

Here we let: the public key cryptosystem be secure against chosen-ciphertext attacks, $E_K()$ be a pseudorandom function and secure against chosen-plaintext attacks, and MAC be a secure message authentication code. We assume that the initialisation is done separately and securely. Also note that a one-way hash function is applied to a message before it is signed.

Step 0: Initialisation: A computes $+K_a, -K_a$. B computes $+K_b, -K_b$. And (as an assumption) they both possess a shared secret, K.

$+K_i$: public key of i,

$-K_i$: private key of i

Step 1: The initiator/requester B chooses a sequence number $seq \xleftarrow{R} \{0,1\}^n$ and a random nonce $N_b \xleftarrow{R} \{0,1\}^n$, where n is 32 bits. B computes $C = E_K(+K_b, N_b, seq)$ and $MAC(C, K)$, and sends $(Req, C, MAC(C, K))$ to A.

Step 2: Upon the receipt of $(Req, C, MAC(C, K))$, A checks the message integrity and computes $m = D_K(C)$. If both are successful then A accepts m and should be able to "make some sense" of the message. A increments seq by 1. A chooses a random challenge $R_A \xleftarrow{R} \{0,1\}^d$ and a random nonce $N_a \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits. A sends $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$ together with his signature $SIG_A(N_a, N_b, seq, R_A, +K_a, B)$ to B. A computes $C = E_K(R_A)$.

Step 3: Upon the receipt of $E_{+K_b}(N_a, N_b, seq, R_A, +K_a)$ and A's signature. B computes $m = D_{-K_b}(E_{+K_b}(N_a, N_b, seq, R_A, +K_a))$ and verifies the signature. If the signature is valid then B verifies the nonce N_b and the sequence number. If they are what B expects then B knows that A also possesses the shared secret K. B increments seq by 1. B computes $C' = E_K(R_A)$ and sends $E_{+K_a}(N_a, seq, C', \beta = g^y)$ together with his signature $SIG_B(N_a, seq, C', \beta = g^y, A)$ to A. B deletes R_A. Note that β is the Diffie-Hellman public component of B.

Step 4: Upon receipt of $E_{+K_a}(N_a, seq, C', \beta)$ and B's signature, A computes $m = D_{-K_a}(E_{+K_a}(N_a, seq, C', \beta))$ and verifies the signature. If the verification succeeds then A verifies the sequence number, the nonce N_a, and checks if $C = C'$. If all are correct, i.e. the values are what A expects, then A increments seq by 1 and sends to B $E_{+K_b}(N_b, seq, \alpha = g^x)$ together with his signature $SIG_A(N_a, seq, \alpha = g^x, \beta, B)$. A is now able to compute the pairwise session key $K_{ab} = \beta^x$. A erases R_A and x. Note that α is the Diffie-Hellman public component of A.

Step 5: Upon receipt of $E_{+K_b}(N_b, seq, \alpha)$ and A's signature, B computes $m = D_{-K_b}(E_{+K_b}(N_b, seq, \alpha))$ and verifies the signature. If the signature is valid then B verifies the sequence number and nonce N_b. If successful then B computes the session key $K_{ab} = \alpha^y$. B erases y and R_A.

A and B have now successfully authenticated one another, and established a new pairwise session key, K_{ab}. Now that an authentication has been accomplished, there is a change in network topology, i.e. B has become a new member of the ad hoc network. By a mechanism of most routing protocols, when the network topology changes, routing information and topology table have to be updated. This is usually done by the exchange of Hello messages among mobile nodes. We extend this mechanism in our pre-handoff protocol so that as a result of the process, mobile nodes will be ready for any handoff process that may take place in the future. This is explained in the next section.

Figure 4-1 summarises the authentication and key establishment protocol described above.

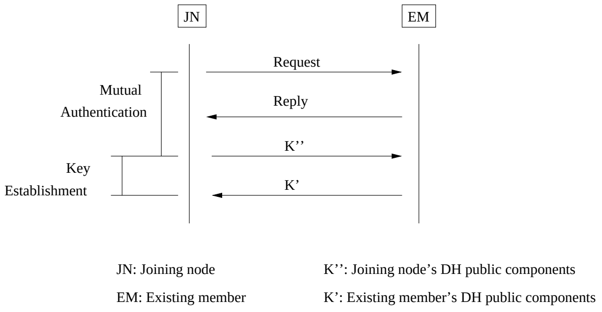


Figure 4-1: Summary of the authentication and key establishment protocol

4.2 Pre-Handoff

Let us say that a node, X, has been authenticated and has joined a mobile ad hoc network (using the process described above). Using the mechanisms of any routing protocol, X can now create a list/table of his one-hop neighbours and generate his topological information. Each of X's neighbours will now send him a *Hello* message containing a list of his one-hop neighbours (X's two-hop neighbours). Once the Hello messages are received, X updates his topology table, and a list of his two-hop neighbours can be created. The purpose of the Hello message is to give the node an overview of the up-to-date topology of the network. The Hello message is used for this same purpose anyway by any routing protocols, particularly the table-driven routing protocols such as link state [CJ03] and fisheye routing [PGC00] protocols.

The following is an extension to the usual exchange of Hello messages. Before a handoff process takes place, the node X will want to make sure that all of his two-hop neighbours hold his security information, namely the RSA public components. X will also want to hold the public key components of each of his two-hop neighbours. The possession of the security information is necessary for achieving fast handoff, because when X moves into the coverage of any of the two-hop neighbours, X will not need to be fully re-authenticated. This is *pre-authentication*, which is under the observation that once you have completed the full authentication process, you do not need to do it again. Also we have designed the protocol in such a way that all of the two-hop neighbours need to hold the security information of X under the assumption that a node always moves/roams past or to its neighbour's neighbour(s).

Instead of "asking" all of his one-hop neighbours to pass the "security information" to the two-hop neighbours, X selects a set of nodes to relay the message. This is done in order to reduce the number of packet duplications and the traffic in the wireless network. This reduction of the traffic and duplication is necessary due to the bandwidth limitations that a mobile ad hoc network has. The algorithm that will be applied to select the relaying nodes is called the *Multipoint Relay* algorithm [QVL00]. The mobile nodes that are selected will be called the *Multipoint Relays* or MPRs. The (formal) algorithm is as follows:

"To select the multipoint relays for the node X, let's call the set of one-hop neighbours of node X as $N(X)$, and the set of its two-hop neighbours as $N^2(X)$.

Let the selected multipoint relay set of node X be $MPR(X)$.

1. Start with an empty multipoint relay set, $MPR(X) = \{\}$.
2. First select those one-hop neighbour nodes in $N(X)$ as the multipoint relays which are the only neighbour of some node in $N^2(X)$, and add these one-hop neighbour nodes to the multipoint relay set.
3. While there still exists some node in $N^2(X)$ which is not covered by the multipoint relay set:
   (a) For each node in $N(X)$ which is not in $MPR(X)$, compute the number of nodes that it covers among the uncovered nodes in the set $N^2(X)$.
   (b) Add that node of $N(X)$ in $MPR(X)$ for which this number is maximum." [QVL00]

Now that X has selected his multipoint relays, he will need to ask them to pass his security information to his two-hop neighbours. Remember that when joining the network, X sent a request packet, including his public key components, to all of his neighbours. That means that the MPRs are already holding X's public key components, `n` and `e`, where `n` is a product of two large prime numbers and `e` is a small odd integer. X asks each multipoint relay to send the public key component, `n` and `e`, to its neighbours. Having received X's public key components, the two-hop neighbours ask the multipoint relays to send their public key components to X. These public key components are needed for the identification process when X roams into the coverage of any of the two-hop neighbours.

In addition, after each mobile node has received the security information of his two-hop neighbours, he will have to compute a new set of Diffie-Hellman components, which are an integral part of the handoff protocol (as we will see in the next section). This step is necessary for achieving a fast handoff. In other words, by pre-calculating the Diffie-Hellman components at the pre-handoff stage, when a handoff process takes place there will not be any need for the travelling mobile node to carry out this computation, which could result in a delay during the handoff.

Now, all the necessary information is being held by appropriate parties, who will be ready for the handoff process whenever it occurs.

4.3 Handoff protocol

We have already stated before that a lot of work has been done on the handoff process in non-pure mobile ad hoc networks. The existing secure handoff protocols only apply to the wireless environment where there is some involvement of home and foreign agents, access points, or some extra third parties. A lot of the work is concerned with the problem of macro-mobility management (movement between domains) instead of the problem of micro-mobility management (movement within one domain), which is what our work focuses on. Furthermore, after having extensively searched through published materials, no research has been found which addresses the problem and security of a roaming or handoff process of a mobile node in pure ad hoc networks. The mechanism described here is concerned with the efficiency and security of roaming process in local (one domain) private mobile ad hoc networks (without any use of the third party agents).

4.3.1 Design

There are four main goals that the handoff protocol needs to achieve. They are:

- make-before-break;
- efficiency, i.e. fast and no noticeable delay;
- pairwise session key establishment with the new node;
- mutual authentication;

Make-before-break means that we would like the roaming node to be able to have made a connection with the node he is approaching before losing the connection with the node he is moving away from. This leads to the efficiency of the protocol. When the handoff protocol is executed, there must not be any long delay. Any delay would be noticeable (a delay of approximately 50 to 100 milliseconds would be considered noticeable) if there are real time or multimedia data involved or waiting to be processed. Therefore, the handoff protocol must be done as fast as possible. Thirdly, pairwise session key establishment with the node he is approaching is important because the new key will provide privacy for the communication between the two parties. The final goal that we would like to achieve is mutual authentication. It is important for this criteria to be satisfied, because when a mobile node moves to a new position, he and the new node need to be sure that the other party can be trusted.

The handoff process will begin when a mobile node A is moving away from a node B towards another mobile node C. As A is moving away from B, the signal received from B becomes weaker. On the other hand, as A is approaching C, the signal from C becomes stronger. There will come a point where the signal from B is below a specified signal threshold, and the signal from C is stronger than that from B and higher than the specified threshold. That is when the exchange of the messages for the actual handoff process is invoked. Note that the strength of the signal can be measured from the Hello message, which is sent periodically by a mobile node as a specification of most routing protocols. Once the threshold of the signal strength is reached, A generates a random token (which will be used in the disassociation process later) and sends it to B.

Having done that A sends a *request-to-roam* packet to C. C then decides whether or not to allow A to roam. C bases his decision on the existence of A, i.e. C checks if A is already a member of the ad hoc network. That is C "decides" whether A has already been pre-authenticated. This can be done by looking up a table/list of existing members (or a topology table) of the ad hoc network which is part of any routing protocol anyway. C also verifies A's identity (further) using digital signature (public-key cryptosystem). A does the same with C's identity. If A is already a member of the network, C and A begin the message exchange, and also the process of pairwise session key establishment. For this purpose, we decide to use the simple two-pass Diffie-Hellman method. (This part of the protocol, i.e. identification and session key establishment, is solely concerned with securing the roaming process. This part also completes mutual authentication.)

After the key has been established, A sends the (disassociation) token to B, via C (because the node A is outside the coverage of the node B). When B receives the disassociation packet, B checks if the token and A's identity match any of the entries in his disassociation table. If so, B removes A from his one-hop neighbour list, indicating that A has now moved away.

4.3.2 Analysis and construction

A handoff protocol is constructed, analysed and proved that it is secure in unauthenticated-links model by following the BCK modular approach [BCK98a, BCK98b]. We also apply the CK analysis method [CK01] to make sure that the communication channel between two parties is secure. The GNY protocol [GNY90] is then used to prove the correctness of the resultant protocol.

As for the authentication and key establishment protocol, we would like to refer the reader to Section 4.1.2 for the theorems and formal definitions that are essential for the analyses and proofs of the handoff protocol.

We divide the handoff protocol into three parts. First, the roaming mobile host sends a random disassociation token to the node he is moving away from. The second part is the actual handoff process. The disassociation process completes the protocol. At the end of the GNY analysis for this design, we will have illustrated that even though the BCK and CK analyses show that the protocol is secure, the GNY analysis proves otherwise. This is the reason that we need to use the BCK and CK methods as well as the GNY protocol to prove that the protocol is both secure and correct. We re-design the insecure part of the handoff protocol in the following section.

We use the following notations throughout this section.

- E_{+K_i} - an asymmetric encryption function with i's public key
- D_{-K_i} - an asymmetric decryption function with i's private key
- $E_{K_{ij}}()$ - a symmetric encryption function with key K shared between party i and party j
- $D_{K_{ij}}()$ - a symmetric decryption function with key K shared between party i and party j
- $MAC()$ - a message authentication code function [KBC97]
- N_i - a random nonce generated by i

<u>Part 1</u> - Pre-disassociation

In this part, A sends a disassociation token to B, who waits until A asks to be disassociated after completing a handoff.

Step 0: A generates a random token $tok \xleftarrow{R} \{0,1\}^d$, a random sequence number $seq \xleftarrow{R} \{0,1\}^n$, and a random nonce (to be used as an identity to which the random token is associated), $id \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits.
Step 1: A computes $C = E_{K_{ab}}(seq, id, tok)$ and $MAC(C, K_{ab})$, and sends $(preDis, C, MAC(C, K_{ab}))$ to B, where K_{ab} is the key shared between A and B and $preDis$ means pre-disassociation. Note that the key was already established earlier as part of the authentication protocol.
Step 2: Upon receipt of $(preDis, C, MAC(C, K_{ab}))$, B checks the message integrity and computes $m = D_{K_{ab}}(C)$. If both are successful then B holds the random token, tok, and the identity, id.

Lemma 4.3.1. *Assume that $E_{K_{ab}}()$ is secure against chosen-plaintext attacks and the MAC scheme is secure. Then we claim that the above steps constitute a secure channel.*

Proof. One message is exchanged in this pre-disassociation part. By Theorem 1 in [Kra01], we can see that a secure channel is achieved, due to the way the message is constructed, namely encrypt-then-authenticate.

By Definition 15 in [CK01], a *secure network channel*, we can conclude that since the channel is secure, a secure network encryption protocol as well as a secure network authentication protocol are achieved. Moreover, this implies that the network channel, and therefore, this part of the protocol is secure in the unauthenticated-links model. □

Part 2 - Handoff process, mutual authentication
This is where the actual handoff process occurs. The mobile nodes that are involved in the handoff process also authenticate one another.

The two-pass Diffie-Hellman protocol is proved to be SK-secure in the authenticated-links model by Canetti and Krawczyk, as stated in Theorem 8 of [CK01].

In order to make the exchange of the messages in the handoff process secure in the unauthenticated-links model, we apply the signature-based MT-authenticator, λ_{sig} [BCK98a, BCK98b], to the protocol packets. By Definition 2 in [BCK98a, BCK98b], we can see that the resultant protocol $\pi' = C_{\lambda_{sig}}(\pi)$ is secure in unauthenticated networks.

Below is the protocol we get after applying the signature-based MT-authenticator, λ_{sig}. Formally, let π be the "original" protocol that is secure in the authenticated-links model (the two-pass Diffie-Hellman in this case). Let π' be the resultant protocol that is secure in unauthenticated networks. We have $\pi' = C_{\lambda_{sig}}(\pi)$:

Step 0: A receives a Hello message from C, whose signal is stronger than that of B.
Step 1: A chooses a random nonce $N_a \xleftarrow{R} \{0,1\}^n$ and a random sequence number $seq \xleftarrow{R} \{0,1\}^n$, where n is 32 bits. A sends the message $E_{+K_c}(Req, N_a, R, seq, \beta = g^y)$ together with his signature $SIG_A(Req, N_a, R, seq, \beta = g^y, C)$, where β is the Diffie-Hellman public component of A which has already been pre-computed during the pre-handoff protocol. Note that R is some random bits that is sent by C in the *reserved* field of the Hello message (or we could add a new field to the Hello message format to hold the random bits), and A possesses C's public key, $+K_c$, due to what happened prior to the handoff process, i.e. the pre-handoff protocol.
Step 2: Upon the receipt of $E_{+K_c}(Req, N_a, R, seq, \beta)$ and A's signature, C computes $m = D_{-Kc}(E_{+K_c}(Req, N_a, R, seq, \beta))$ and verifies the signature. (We will discuss the verification of R later.) C checks his topology table to see if A has already been pre-authenticated, or if A is already a member of the network. If the signature is valid and A is already an existing member then C increments seq by 1 and sends to A, $E_{+K_a}(N_a, seq, \alpha = g^x)$ together with his signature $SIG_C(N_a, seq, \alpha = g^x, A)$, where α is the Diffie-Hellman public component of C. C is now able to compute the pairwise session key $K_{ac} = \beta^x$. C also possesses A's public key, $+K_a$, as a result of the pre-handoff protocol.
Step 3: Upon the receipt of $E_{+K_a}(N_a, seq, \alpha)$ and C's signature, A computes $m = D_{-K_a}(E_{+K_a}(N_a, seq, \alpha))$ and verifies the signature. (We will talk about the nonce and sequence number verification later.) If the signature is valid then A computes the session key $K_{ac} = \alpha^y$.

Lemma 4.3.2. *The above protocol is secure in the unauthenticated-links model.*

Proof. To show that this protocol is secure in the unauthenticated-links model, we need to illustrate that the protocol conforms to the signature-based MT-authenticator, λ_{sig}.

In step 1, we use R, a random value sent as part of the Hello message as a challenge[5]. Note that at the moment, in the Hello message in any routing protocols, there is a reserved field, which is unused. We, therefore, propose that a random value should be put in that field. The use of R as a challenge is adequate since it is chosen freshly and at random.

In step 2, C only accepts the packet if the verification of the signature is successful. In the previous step, A sent to C a random nonce N_a, which can be used as a challenge required by the authenticator. C then sends a message to A together with his signature, which includes the challenge, N_a.

In the final step, the message from C is accepted if the signature verification is a success.

It appears that our protocol behaves as required by the signature-based MT-authenticator, therefore, we can claim, by Definition 2 in [BCK98a], that it is secure in the unauthenticated-links model. □

<u>Part 3</u> - Disassociation
Here, the roaming node informs another node that he has already moved away.
Step 0: A holds the random token, *tok*, the sequence number, *seq*, and the random nonce, *id*, from Step 0 in Part 1.
Step 1: A increments *seq* by 1. A computes $C = E_{K_{ab}}(seq, id, tok)$ and $MAC(C, K_{ab})$. A sends $(dis, C, MAC(C, K_{ab}))$ to C, the node he has just roamed to, where *dis* means disassociation.
Step 2: Upon receipt of $(dis, C, MAC(C, K_{ab}))$, C looks at the destination address and forwards the message to B. Bear in mind that routing or forwarding packets is part of mobile ad hoc networks anyway.
Step 3: Upon receipt of $(dis, C, MAC(C, K_{ab}))$, whose source address is A, B checks the message integrity and computes $m = D_{K_{ab}}(C)$. If B's calculations are successful, B checks the sequence number. B then compares the received *id* and *tok* with the entries in his table. If the values match, then B disassociates A, indicating that A has now moved away and is no longer one of B's one-hop neighbours.

Lemma 4.3.3. *Assume that $E_{K_{ab}}()$ is secure against chosen-plaintext attacks and the MAC scheme is secure. Then we claim that the above steps constitute a*

[5] A challenge is an integral part of the signature-based MT-authenticator, λ_{sig}.

secure channel.

Proof. Only one message is sent from A to B, albeit via C. By Theorem 1 in [Kra01], a secure channel is accomplished because of the way the packet is constructed, i.e. encrypt-then-authenticate.

By Definition 15 in [CK01], a *secure network channel*, we can conclude that if the channel is secure then a secure network encryption protocol as well as a secure network authentication protocol are achieved. Moreover, this implies that the network channel, and therefore, this part of the protocol is secure in the unauthenticated-links model. □

4.3.3 Proof of correctness

We will analyse the resultant protocol of the handoff process by using the GNY logic. The notations and logical postulates of the GNY protocol that are applied here can be found in Appendix A. The protocol written in the GNY logic is as follows:

1. A→B: B: $\triangleleft\ *preDis, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})$, where *preDis* means pre-disassociation.

2. A→C: C: $\triangleleft *\{*Req, *N_a, R, *seq, *K''\}_{+K_c}, *\{*H(*Req, *N_a, R, *seq, *K'', C)\}_{-K_a}$ $\leadsto A \mid\equiv A \overset{K''}{\longleftrightarrow} C$, where $K'' = g^y \bmod p$

3. C→A: A: $\triangleleft *\{N_a, *seq, *K'\}_{+K_a}, *\{*H(N_a, *seq, *K', A)\}_{-K_c}$ $\leadsto C \mid\equiv C \overset{K'}{\longleftrightarrow} A$, where $K' = g^x \bmod p$

4. A→C: C: $\triangleleft *\{*dis, *id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})$, where *dis* means disassociation.

5. C→B: B: $\triangleleft *dis*\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})$, where *dis* means disassociation.

Assumptions

$\mathrm{A}|\!\equiv\!\mathrm{A}\overset{K_{ab}}{\longleftrightarrow}\mathrm{B}$ $\mathrm{B}|\!\equiv\!\mathrm{B}\overset{K_{ab}}{\longleftrightarrow}\mathrm{A}$ $\mathrm{A}|\!\equiv\!\mathrm{A}\overset{K''}{\longleftrightarrow}\mathrm{C}$

$\mathrm{C}|\!\equiv\!\mathrm{C}\overset{K'}{\longleftrightarrow}\mathrm{A}$ $\mathrm{A}|\!\equiv\!\mathrm{C}|\!\Longrightarrow\!\mathrm{C}\overset{K'}{\longleftrightarrow}\mathrm{A}$ $\mathrm{C}|\!\equiv\!\mathrm{A}|\!\Longrightarrow\!\mathrm{A}\overset{K''}{\longleftrightarrow}\mathrm{C}$

$\mathrm{A}|\!\equiv\!\mathrm{C}|\!\Longrightarrow\!\mathrm{C}|\!\equiv *$ $\mathrm{C}|\!\equiv\!\mathrm{A}|\!\Longrightarrow\!\mathrm{A}|\!\equiv *$ $\mathrm{A}|\!\equiv \sharp(N_a)$

$\mathrm{A}|\!\equiv \sharp(id)$ $\mathrm{A}|\!\equiv \sharp(tok)$ $\mathrm{C}|\!\equiv \sharp(R)$

$\mathrm{A}\ni +K_a$ $\mathrm{C}\ni +K_c$ $\mathrm{A}\ni -K_a$

$\mathrm{C}\ni -K_c$ $\mathrm{A}\ni N_a$ $\mathrm{A}\ni id$

$\mathrm{A}\ni tok$ $\mathrm{C}\ni R$ $\mathrm{A}\ni +K_c$

$\mathrm{C}\ni +K_a$ $\mathrm{A}\ni K_{ab}$ $\mathrm{B}\ni K_{ab}$

The final four are not really assumptions since A, B and C do really possess each other's public key, because they are the results of the pre-handoff stage.

Notations:

- $+K_a$ - A's public key
- $-K_a$ - A's private key
- K'' - A's DH public component
- K_{ab} - A and B's pairwise session key
- $+K_c$ - C's public key
- $-K_c$ - C's private key
- K' - C's DH public component
- $H()$ - a one-way hash function

Analysis

Message 1:

T1: $$\frac{B \lhd *preDis, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})}{B \lhd preDis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}})}$$

T3: $$\frac{B \lhd preDis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}}), B \ni K_{ab}}{B \lhd preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

P1: $$\frac{B \lhd preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}{B \ni preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

Since $B \ni id, seq, tok$ and $B \ni K_{ab}$, by the rule P4, $B \ni H(\{id, seq, tok\}_{K_{ab}})$. Now B can check if $H(\{id, seq, tok\}_{K_{ab}}$ is equal to $MAC(\{id, seq, tok\}_{K_{ab}})$. If so, B knows that the message has not been tampered with.

Message 2:

$A \mid\equiv A \overset{K''}{\longleftrightarrow} C$ is valid because it is an initial assumption.

$$\text{T1: } \frac{C \triangleleft *\{*Req, *N_a, R, *seq, *K''\}_{+K_c}, *\{*H(*Req, *N_a, R, *seq, *K'', C)\}_{-K_a}}{C \triangleleft \{Req, N_a, R, seq, K''\}_{+K_c}, \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}$$

$$\text{T4: } \frac{C \triangleleft \{Req, N_a, R, seq, K''\}_{+K_c}, \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}, C \ni -K_c}{C \triangleleft Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}$$

$$\text{P1: } \frac{C \triangleleft Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}{C \ni Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}$$

$$\text{P4: } \frac{C \ni (Req, N_a, R, seq, K''), C \ni C}{C \ni H(Req, N_a, R, seq, K'', C)}$$

Note that $C \ni C$ because C possesses his own identity anyway.

$$\text{F1: } \frac{C \mid\equiv \sharp(N_a, seq)}{C \mid\equiv \sharp(Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a})}$$

$$\text{R1: } \frac{C \mid\equiv \phi R}{C \mid\equiv \phi(Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a})}$$

Let $(*Req, *N_a, R, *seq, *K'')$ be X.

$$\text{R5: } \frac{C \mid\equiv \phi(X), C \ni (X, C), C \mid\equiv \phi(C)}{C \mid\equiv \phi(H(X)), C \mid\equiv \phi(H(X, C))}$$

$$\text{I4: } \frac{C \triangleleft \{H(X, C)\}_{-K_a}, C \ni +K_a, C \mid\equiv \overset{+K_a}{\mapsto} A, C \mid\equiv \phi(H(X, C))}{C \mid\equiv A \mid\sim H(X, C), C \mid\equiv A \mid\sim \{H(X, C)\}_{-K_a}}$$

C believes that A signed the message, and therefore, sent the message. That is $C \mid\equiv A \mid\sim X, H(X, C)$.

$$\text{J2: } \frac{C \mid\equiv A \mid\Longrightarrow A \mid\equiv *, C \mid\equiv A \mid\sim (X \leadsto A \mid\equiv A \xleftrightarrow{K''} C), C \mid\equiv \sharp(X)}{C \mid\equiv A \mid\equiv A \xleftrightarrow{K''} C}$$

$$\text{J1: } \frac{C \mid\equiv A \mid\Longrightarrow A \xleftrightarrow{K''} C, C \mid\equiv A \mid\equiv A \xleftrightarrow{K''} C}{C \mid\equiv A \xleftrightarrow{K''} C}$$

which now implies that $C \mid\equiv A \xleftrightarrow{K_{ac}} C$. After this message the party C believes that the freshly generated key, K_{ac}, is shared between himself/herself and the party A.

Message 3:

$C \mid\equiv C \xleftrightarrow{K'} A$ is valid because it is an initial assumption.

$$\text{T1: } \frac{A \lhd *\{N_a, *seq, *K'\}_{+K_a}, *\{H(N_a, *seq, *K', A)\}_{-K_c}}{A \lhd \{N_a, seq, K'\}_{+K_a}, \{H(N_a, seq, K', A)\}_{-K_c}}$$

$$\text{T4: } \frac{A \lhd \{N_a, seq, K'\}_{+K_a}, \{H(N_a, seq, K', A)\}_{-K_c}, A \ni -K_a}{A \lhd N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}$$

$$\text{P1: } \frac{A \lhd N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}{A \ni N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}$$

$$\text{P4: } \frac{A \ni (N_a, seq, K'), A \ni A}{A \ni H(N_a, seq, K', A)}$$

Note that $A \ni A$ because A possesses his own identity anyway.

$$\text{F1: } \frac{A \mid\equiv \sharp seq}{A \mid\equiv \sharp(N_a, seq, K', \{N_a, seq, K', A\}_{-K_c})}$$

$$\text{R1: } \frac{A \mid\equiv \phi N_a}{A \mid\equiv \phi(N_a, seq, K', \{N_a, seq, K', A\}_{-K_c})}$$

Let $(N_a, *seq, *K')$ be X.

$$\text{R5: } \frac{A \mid\equiv \phi(X), A \ni (X, A), A \mid\equiv \phi(A)}{A \mid\equiv \phi(H(X)), A \mid\equiv (H(X, A))}$$

$$\text{I4: } \frac{A \lhd \{H(X, A)\}_{-K_c}, A \ni +K_c, A \mid\equiv \overset{+K_c}{\mapsto} C, A \mid\equiv \phi(H(X, A))}{A \mid\equiv C \mid\sim H(X, A), A \mid\equiv C \mid\sim \{H(X, A)\}_{-K_c}}$$

A believes that C signed the message, and therefore, sent the message. That is $A \mid\equiv C \mid\sim X, H(X, A)$.

$$\text{J2: } \frac{A \mid\equiv C \mid\Longrightarrow C \mid\equiv *, A \mid\equiv C \mid\sim (X \leadsto C \mid\equiv C \xleftrightarrow{K'} A), B \mid\equiv \sharp(X)}{A \mid\equiv C \mid\equiv C \xleftrightarrow{K'} A}$$

$$\text{J1: } \frac{A \mid\equiv C \mid\Longrightarrow C \xleftrightarrow{K'} A, A \mid\equiv C \mid\equiv C \xleftrightarrow{K'} A}{A \mid\equiv C \xleftrightarrow{K'} A}$$

which now implies that $A \mid\equiv C \overset{K_{ac}}{\longleftrightarrow} A$. At this stage, A believes that the key, K_{ac}, is a secret shared between that parties A and C.

Therefore, we get:

$$\begin{array}{lll} C \mid\equiv A \overset{K''}{\longleftrightarrow} C & \text{and} & A \mid\equiv C \overset{K'}{\longleftrightarrow} A \\ C \mid\equiv A \mid\equiv A \overset{K''}{\longleftrightarrow} C & & A \mid\equiv C \mid\equiv C \overset{K'}{\longleftrightarrow} A \end{array}$$

which imply that

$$\begin{array}{lll} C \mid\equiv A \overset{K_{ac}}{\longleftrightarrow} C & \text{and} & A \mid\equiv C \overset{K_{ac}}{\longleftrightarrow} A \\ C \mid\equiv A \mid\equiv A \overset{K_{ac}}{\longleftrightarrow} C & & A \mid\equiv C \mid\equiv C \overset{K_{ac}}{\longleftrightarrow} A \end{array}$$

where $K_{ac} = (g^x)^y \bmod p = (g^y)^x \bmod p$.

At the end of message 3, we have achieved the following outcomes. The parties A and C believe that the key, K_{ac}, is shared between themselves. Each of them also believes that the other party believes that the key, K_{ac}, is a suitable secret for A and C.

Message 4:

$$\text{T1: } \frac{C \lhd *dis, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})}{C \lhd dis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}})}$$

C, having received this message, forwards it to B.

Message 5:

$$\text{T1: } \frac{B \lhd *dis, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})}{B \lhd dis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}})}$$

$$\text{T3: } \frac{B \lhd dis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}}), B \ni K_{ab}}{B \lhd dis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

$$\text{P1: } \frac{B \lhd dis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}{B \ni dis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

Since $B \ni id, seq, tok$ and $B \ni K_{ab}$, by the rule P4, $B \ni H(\{id, seq, tok\}_{K_{ab}})$. Now B can check if $H(\{id, seq, tok\}_{K_{ab}}$ is equal to $MAC(\{id, seq, tok\}_{K_{ab}})$. If so, B knows that the message has not been tampered with.

F1: $\dfrac{B \mid\equiv \sharp seq}{B \mid\equiv \sharp(dis, id, seq, tok)}$

Applying the rule R1, the recognisability rule, the process does not terminate. This is due to the fact that there is nothing that the node B recognises from earlier exchange of messages. B, therefore, cannot be certain that this message is really from the party that he has been communicating with, i.e. the node A.

The GNY analysis has shown that potentially the disassociation stage can be under threat. In order to solve the problem of recognisability, we need to go back to the pre-disassociation stage. The node B, after receiving the disassociation token (*tok*) from A, will have to send a "reply" or an acknowledgement packet to A in order to indicate that the token has been received.

4.3.4 Re-design

The following is the description of the re-designed pre-disassociation part of the protocol. It will be analysed and proved to make sure that it is secure and correct.

Part 1 - Pre-disassociation

In this part, A sends a disassociation token to B, who waits until A asks to be disassociated after completing a handoff. B informs A that he has received the token.

Step 0: A generates a random token $tok \xleftarrow{R} \{0,1\}^d$, a random sequence number $seq \xleftarrow{R} \{0,1\}^n$, and a random nonce (to be used as an identity to which the random token is associated) $id \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits.

Step 1: A computes $C = E_{K_{ab}}(seq, id, tok)$ and $MAC(C, K_{ab})$, and sends $(preDis, C, MAC(C, K_{ab}))$ to B, where *preDis* means pre-disassociation.

Step 2: Upon receipt of $(preDis, C, MAC(C, K_{ab}))$, B checks the message integrity and computes $m = D_{K_{ab}}(C)$. If both are successful then B holds the random token, *tok*, and the identity, *id*. B chooses a random nonce, $N_b \xleftarrow{R} \{0,1\}^n$, where n is 32 bits, and increments the sequence number by 1. B computes $C = E_{K_{ab}}(id, seq, Nb)$ and $MAC(C, K_{ab})$. B then sends to A $(Ack, C, MAC(C, K_{ab}))$.

Step 3: Upon receipt of $(Ack, C, MAC(C, K_{ab}))$, A checks the message integrity and computes $m = D_{K_{ab}}(C)$. If all are successful then A knows that B has received the message. A saves the random nonce, N_b.

Lemma 4.3.4. *Assume that $E_{K_{ab}}()$ is secure against chosen-plaintext attacks and the MAC scheme is secure. Then we claim that the above steps constitute a secure channel.*

Proof. Two messages are exchanged in this re-designed pre-disassociation part of the protocol. The two packets are constructed in the same way, namely encrypt-then-authenticate. By Theorem 1 in [Kra01], we can see that a secure channel is achieved.

By Definition 15 in [CK01], a *secure network channel*, we can conclude that if the channel is secure then a secure network encryption protocol as well as a secure network authentication protocol are accomplished. Moreover, this implies that the network channel, and therefore, this part of the protocol, which includes the exchange of two messages, is secure in the unauthenticated-links model. □

We have shown that the newly re-designed pre-disassociation part is secure in the unauthenticated networks. The pre-disassociation module will be integrated into the rest of the unchanged protocol. We will carry out the GNY analysis again in order to make sure that the proof of correctness terminates, hence potential attacks are no longer possible.

4.3.5 Proof of correctness (again)

We analyse the protocol with the newly integrated pre-disassociation part. Appendix A explains the notations and rules of the GNY protocol that are applied here. The handoff protocol can be written in the GNY logic as follows:

1. A→B: B: $\triangleleft *\mathit{preDis}, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})$, where *preDis* means pre-disassociation.

2. B→A: A: $\triangleleft *Ack, *\{id, *seq, *N_b\}_{K_{ab}}, *MAC(*\{id, *seq, *N_b\}_{K_{ab}})$, where *Ack* means acknowledgement.

3. A→C: C: $\triangleleft *\{*Req, *N_a, R, *seq, *K''\}_{+K_c}, *\{*H(*Req, *N_a, R, *seq, *K'', C)\}_{-K_a}$ $\leadsto A \mid\equiv A \xleftrightarrow{K''} C$, where $K'' = g^y \bmod p$

4. C→A: A: $\triangleleft *\{N_a, *seq, *K'\}_{+K_a}, *\{*H(N_a, *seq, *K', A)\}_{-K_c}$ $\leadsto C \mid\equiv C \xleftrightarrow{K'} A$, where $K' = g^x \bmod p$

5. A→C: C: $\lhd *dis, *\{*id, *seq, *N_b, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *N_b, *tok\}_{K_{ab}})$, where *dis* means disassociation.

6. C→B: B: $\lhd *dis, *\{*id, *seq, N_b, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, N_b, *tok\}_{K_{ab}})$, where *dis* means disassociation.

Assumptions

$\text{A}\vert\equiv\text{A}\xleftrightarrow{K_{ab}}\text{B}$	$\text{B}\vert\equiv\text{B}\xleftrightarrow{K_{ab}}\text{A}$	$\text{A}\vert\equiv\text{A}\xleftrightarrow{K''}\text{C}$
$\text{C}\vert\equiv\text{C}\xleftrightarrow{K'}\text{A}$	$\text{A}\vert\equiv\text{C}\vert\Longrightarrow\text{C}\xleftrightarrow{K'}\text{A}$	$\text{C}\vert\equiv\text{A}\vert\Longrightarrow\text{A}\xleftrightarrow{K''}\text{C}$
$\text{A}\vert\equiv\text{C}\vert\Longrightarrow\text{C}\vert\equiv *$	$\text{C}\vert\equiv\text{A}\vert\Longrightarrow\text{A}\vert\equiv *$	$\text{A}\vert\equiv \sharp(N_a)$
$\text{A}\vert\equiv \sharp(id)$	$\text{A}\vert\equiv \sharp(tok)$	$\text{C}\vert\equiv \sharp(R)$
$\text{A}\ni +K_a$	$\text{C}\ni +K_c$	$\text{A}\ni -K_a$
$\text{C}\ni -K_c$	$\text{A}\ni N_a$	$\text{A}\ni id$
$\text{A}\ni tok$	$\text{C}\ni R$	$\text{A}\ni +K_c$
$\text{C}\ni +K_a$	$\text{A}\ni K_{ab}$	$\text{B}\ni K_{ab}$

The final four are not really assumptions since A, B and C do really possess each other's public key, because they are the results of the pre-handoff protocol.

Notations:

- $+K_a$ - A's public key
- $-K_a$ - A's private key
- K'' - A's DH public component
- K_{ab} - A and B's pairwise session key
- $+K_c$ - C's public key
- $-K_c$ - C's private key
- K' - C's DH public component
- $H()$ - a one-way hash function

Analysis

Message 1:

T1: $$\frac{B \lhd *preDis, *\{*id, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *seq, *tok\}_{K_{ab}})}{B \lhd preDis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}})}$$

T3: $$\frac{B \lhd preDis, \{id, seq, tok\}_{K_{ab}}, MAC(\{id, seq, tok\}_{K_{ab}}), B \ni K_{ab}}{B \lhd preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

P1: $$\frac{B \lhd preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}{B \ni preDis, id, seq, tok, MAC(\{id, seq, tok\}_{K_{ab}})}$$

Since $B \ni id, seq, tok$ and $B \ni K_{ab}$, by the rule P4, $B \ni H(\{id, seq, tok\}_{K_{ab}})$. Now B can check if $H(\{id, seq, tok\}_{K_{ab}})$ is equal to $MAC(\{id, seq, tok\}_{K_{ab}})$. If so, B knows that the message has not been tampered with.

Message 2:

T1: $$\frac{A \lhd *Ack, *\{id, *seq, *N_b\}_{K_{ab}}, *MAC(*\{id, *seq, *N_b\}_{K_{ab}})}{A \lhd Ack, \{id, seq, N_b\}_{K_{ab}}, MAC(\{id, seq, N_b\}_{K_{ab}})}$$

T3: $$\frac{A \lhd Ack, \{id, seq, N_b\}_{K_{ab}}, MAC(\{id, seq, N_b\}_{K_{ab}}), A \ni K_{ab}}{A \lhd Ack, id, seq, N_b, MAC(\{id, seq, N_b\}_{K_{ab}})}$$

P1: $$\frac{A \lhd Ack, id, seq, N_b, MAC(\{id, seq, N_b\}_{K_{ab}})}{A \ni Ack, id, seq, N_b, MAC(\{id, seq, N_b\}_{K_{ab}})}$$

Since $A \ni id, seq, N_b$ and $A \ni K_{ab}$ by the rule P4, $A \ni H(\{id, seq, N_b\}_{K_{ab}})$. Now A can check if $H(\{id, seq, N_b\}_{K_{ab}})$ is equal to $MAC(\{id, seq, N_b\}_{K_{ab}})$. If they equal, A knows that the message is authentic, i.e. it has not been tampered with.

F1: $$\frac{A \mid\equiv \sharp(seq, N_b)}{A \mid\equiv \sharp(Ack, id, seq, N_b)}$$

R1: $$\frac{A \mid\equiv \phi(id)}{A \mid\equiv \phi(Ack, id, seq, N_b)}$$

Let $(id, *seq, *N_b)$ be X.

I1: $$\frac{A \lhd *\{X\}_{K_{ab}}, A \ni K_{ab}, A \mid\equiv A \overset{K_{ab}}{\longleftrightarrow} B, A \mid\equiv \phi(X), A \mid\equiv \sharp(X)}{A \mid\equiv B \mid\sim X, A \mid\equiv B \mid\sim \{X\}_{K_{ab}}, A \mid\equiv B \ni K_{ab}}$$

A believes that B sent the message, hence B had received the disassociation token sent by A in the previous message.

Message 3:

$A \mid\equiv A \overset{K''}{\longleftrightarrow} C$ is valid because it is an initial assumption.

T1: $$\frac{C \lhd *\{*Req, *N_a, R, *seq, *K''\}_{+K_c}, *\{*H(*Req, *N_a, R, *seq, *K'', C)\}_{-K_a}}{C \lhd \{Req, N_a, R, seq, K''\}_{+K_c}, \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}$$

T4: $$\frac{C \lhd \{Req, N_a, R, seq, K''\}_{+K_c}, \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}, C \ni -K_c}{C \lhd Req, N_a, R, seq, K'', \{H(Req, N_a, R, seq, K'', C)\}_{-K_a}}$$

P1: $$\frac{C \triangleleft \mathit{Req}, N_a, R, \mathit{seq}, K'', \{H(\mathit{Req}, N_a, R, \mathit{seq}, K'', C)\}_{-K_a}}{C \ni \mathit{Req}, N_a, R, \mathit{seq}, K'', \{H(\mathit{Req}, N_a, R, \mathit{seq}, K'', C)\}_{-K_a}}$$

P4: $$\frac{C \ni (\mathit{Req}, N_a, R, \mathit{seq}, K''), C \ni C}{C \ni H(\mathit{Req}, N_a, R, \mathit{seq}, K'', C)}$$

Note that $C \ni C$ because C possesses his own identity anyway.

F1: $$\frac{C \mid\equiv \sharp(N_a, \mathit{seq})}{C \mid\equiv \sharp(\mathit{Req}, N_a, R, \mathit{seq}, K'', \{H(\mathit{Req}, N_a, R, \mathit{seq}, K'', C)\}_{-K_a})}$$

R1: $$\frac{C \mid\equiv \phi R}{C \mid\equiv \phi(\mathit{Req}, N_a, R, \mathit{seq}, K'', \{H(\mathit{Req}, N_a, R, \mathit{seq}, K'', C)\}_{-K_a})}$$

Let $(*\mathit{Req}, *N_a, R, *\mathit{seq}, *K'')$ be X.

R5: $$\frac{C \mid\equiv \phi(X), C \ni (X, C), C \mid\equiv \phi(C)}{C \mid\equiv \phi(H(X)), C \mid\equiv \phi(H(X, C))}$$

I4: $$\frac{C \triangleleft \{H(X, C)\}_{-K_a}, C \ni +K_a, C \mid\equiv \overset{+K_a}{\mapsto} A, C \mid\equiv \phi(H(X, C))}{C \mid\equiv A \mid\sim H(X, C), C \mid\equiv A \mid\sim \{H(X, C)\}_{-K_a}}$$

C believes that A signed the message, and therefore, sent the message. That is $C \mid\equiv A \mid\sim X, H(X, C)$.

J2: $$\frac{C \mid\equiv A \mid\Longrightarrow A \mid\equiv *, C \mid\equiv A \mid\sim (X \leadsto A \mid\equiv A \xleftrightarrow{K''} C), C \mid\equiv \sharp(X)}{C \mid\equiv A \mid\equiv A \xleftrightarrow{K''} C}$$

J1: $$\frac{C \mid\equiv A \mid\Longrightarrow A \xleftrightarrow{K''} C, C \mid\equiv A \mid\equiv A \xleftrightarrow{K''} C}{C \mid\equiv A \xleftrightarrow{K''} C}$$

which now implies that $C \mid\equiv A \xleftrightarrow{K_{ac}} C$. After this message the party C believes that the freshly generated key, K_{ac}, is shared between himself/herself and the party A.

Message 4:
$C \mid\equiv C \xleftrightarrow{K'} A$ is valid because it is an initial assumption.

T1: $$\frac{A \triangleleft *\{N_a, *seq, *K'\}_{+K_a}, *\{*H(N_a, *seq, *K', A)\}_{-K_c}}{A \triangleleft \{N_a, seq, K'\}_{+K_a}, \{H(N_a, seq, K', A)\}_{-K_c}}$$

T4: $$\frac{A \triangleleft \{N_a, seq, K'\}_{+K_a}, \{H(N_a, seq, K', A)\}_{-K_c}, A \ni -K_a}{A \triangleleft N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}$$

P1: $$\frac{A \triangleleft N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}{A \ni N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c}}$$

P4: $$\frac{A \ni (N_a, seq, K'), A \ni A}{A \ni H(N_a, seq, K', A)}$$

Note that $A \ni A$ because A possesses his own identity anyway.

F1: $$\frac{A \mid\equiv \sharp seq}{A \mid\equiv \sharp(N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c})}$$

R1: $$\frac{A \mid\equiv \phi N_a}{A \mid\equiv \phi(N_a, seq, K', \{H(N_a, seq, K', A)\}_{-K_c})}$$

Let $(N_a, *seq, *K')$ be X.

R5: $$\frac{A \mid\equiv \phi(X), A \ni (X, A), A \mid\equiv \phi(A)}{A \mid\equiv \phi(H(X)), A \mid\equiv (H(X, A))}$$

I4: $$\frac{A \triangleleft \{H(X, A)\}_{-K_c}, A \ni +K_c, A \mid\equiv \overset{+K_c}{\mapsto} C, A \mid\equiv \phi(H(X, A))}{A \mid\equiv C \mid\sim H(X, A), A \mid\equiv C \mid\sim \{H(X, A)\}_{-K_c}}$$

A believes that C signed the message, and therefore, sent the message. That is $A \mid\equiv C \mid\sim X, H(X, C)$.

J2: $$\frac{A \mid\equiv C \mid\Longrightarrow C \mid\equiv *, A \mid\equiv C \mid\sim (X \leadsto C \mid\equiv C \xleftrightarrow{K'} A), B \mid\equiv \sharp(X)}{A \mid\equiv C \mid\equiv C \xleftrightarrow{K'} A}$$

J1: $$\frac{A \mid\equiv C \mid\Longrightarrow C \xleftrightarrow{K'} A, A \mid\equiv C \mid\equiv C \xleftrightarrow{K'} A}{A \mid\equiv C \xleftrightarrow{K'} A}$$

which now implies that $A \mid\equiv C \xleftrightarrow{K_{ac}} A$. At this stage, A believes that the key, K_{ac}, is a secret shared between that parties A and C.

Therefore, we get:

$$\begin{array}{lcl} C \mid\equiv A \overset{K''}{\longleftrightarrow} C & \text{and} & A \mid\equiv C \overset{K'}{\longleftrightarrow} A \\ C \mid\equiv A \mid\equiv A \overset{K''}{\longleftrightarrow} C & & A \mid\equiv C \mid\equiv C \overset{K'}{\longleftrightarrow} A \end{array}$$

which imply that

$$\begin{array}{lcl} C \mid\equiv A \overset{K_{ac}}{\longleftrightarrow} C & \text{and} & A \mid\equiv C \overset{K_{ac}}{\longleftrightarrow} A \\ C \mid\equiv A \mid\equiv A \overset{K_{ac}}{\longleftrightarrow} C & & A \mid\equiv C \mid\equiv C \overset{K_{ac}}{\longleftrightarrow} A \end{array}$$

where $K_{ac} = (g^x)^y \bmod p = (g^y)^x \bmod p$.

At the end of message 4, we have achieved the following outcomes. The parties A and C believe that the key, K_{ac}, is shared between themselves. Each of them also believes that the other party believes that the key, K_{ac}, is a suitable secret for A and C.

Message 5:

T1: $$\frac{C \lhd *dis, *\{*id, *N_b, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, *N_b, *seq, *tok\}_{K_{ab}})}{C \lhd dis, \{id, N_b, seq, tok\}_{K_{ab}}, MAC(\{id, N_b, seq, tok\}_{K_{ab}})}$$

C, having received this message, forwards it to B.

Message 6:

T1: $$\frac{B \lhd *dis, *\{*id, N_b, *seq, *tok\}_{K_{ab}}, *MAC(*\{*id, N_b, *seq, *tok\}_{K_{ab}})}{B \lhd dis, \{id, N_b, seq, tok\}_{K_{ab}}, MAC(\{id, N_b, seq, tok\}_{K_{ab}})}$$

T3: $$\frac{B \lhd dis, \{id, N_b, seq, tok\}_{K_{ab}}, MAC(\{id, N_b, seq, tok\}_{K_{ab}}), B \ni K_{ab}}{B \lhd dis, id, N_b, seq, tok, MAC(\{id, N_b, seq, tok\}_{K_{ab}})}$$

P1: $$\frac{B \lhd dis, id, N_b, seq, tok, MAC(\{id, N_b, seq, tok\}_{K_{ab}})}{B \ni dis, id, N_b, seq, tok, MAC(\{id, N_b, seq, tok\}_{K_{ab}})}$$

Since $B \ni id, N_b, seq, tok$ and $B \ni K_{ab}$, by the rule P4, $B \ni H(\{id, N_b, seq, tok\}_{K_{ab}})$. Now B can check if $H(\{id, N_b, seq, tok\}_{K_{ab}}$ is equal to $MAC(\{id, N_b, seq, tok\}_{K_{ab}})$. If so, B knows that the message has not been tampered with.

F1: $$\frac{B \mid\equiv \sharp(seq)}{B \mid\equiv \sharp(dis, id, N_b, seq, tok)}$$

R1: $$\frac{B \mid\equiv \phi(N_b)}{B \mid\equiv \phi(dis, id, N_b, seq, tok)}$$

B has received a message with something he recognises. Therefore, with the new design, the rule R1 is satisfied.

Let $(*id, N_b, *seq, *tok)$ be X.

I1: $$\frac{B \lhd *\{X\}_{K_{ab}}, B \ni K_{ab}, B \mid\equiv B \overset{K_{ab}}{\longleftrightarrow} A, B \mid\equiv \phi(X), B \mid\equiv \sharp(X)}{B \mid\equiv A \mid\sim X, B \mid\equiv A \mid\sim \{X\}_{K_{ab}}, B \mid\equiv A \ni K_{ab}}$$

B believes that the message really originated from A.

B is now able to compare the received *id* and *tok* with the ones in his list. If the values match then B disassociates A, indicating that A has now moved and is no longer his one-hop neighbours.

4.3.6 Description of the resultant protocol

We now summarise the handoff protocol that, we believe, is correct and secure in unauthenticated-links model. Again, we use the following notations throughout the description of the protocol.

- E_{+K_i} - an asymmetric encryption function with i's public key
- D_{-K_i} - an asymmetric decryption function with i's private key
- $E_{K_{ij}}()$ - a symmetric encryption function with key K shared between party i and party j
- $D_{K_{ij}}()$ - a symmetric decryption function with key K shared between party i and party j
- $MAC()$ - a message authentication code function [KBC97]
- N_i - a random nonce generated by i

Here we let: the public key cryptosystem be secure against chosen-ciphertext attacks, $E_K()$ be secure against chosen-plaintext attacks, and MAC be a secure message authentication code. Also note that a one-way hash function is applied to a message before it is signed.

Step 0: A receives a Hello message from C, whose signal is stronger than that of B. Within the Hello message, there is a field that contains a random *challenge.* A holds his own public and private keys, $+K_a$ and $-K_a$, and also, from the earlier exchange of messages during the pre-handoff protocol (see Section 4.2) A knows C's public key, $+K_c$. Similarly, C holds his own public and private key pair, $+K_c$ and $-K_c$, and also, as a result of the pre-handoff protocol, A's public key, $+K_a$ is known.

A generates a random token $tok \xleftarrow{R} \{0,1\}^d$, a random sequence number $seq \xleftarrow{R} \{0,1\}^n$, and a random nonce (to be used as an identity to which the random token is associated) $id \xleftarrow{R} \{0,1\}^n$, where d is 256 bits and n is 32 bits.

$+K_i$: public key of i,

$-K_i$: private key of i,

K_{ij}: session key of i and j

Step 1: A computes $Ciph = E_{K_{ab}}(seq, id, tok)$ and $MAC(Ciph, K_{ab})$, and sends $(preDis, Ciph, MAC(Ciph, K_{ab}))$ to B, where *preDis* means pre-disassociation.

Step 2: Upon receipt of $(preDis, Ciph, MAC(Ciph, K_{ab}))$, B checks the message integrity and computes $m = D_{K_{ab}}(Ciph)$. If both are successful then B holds the random token, *tok*, and the identity, *id.* B chooses a random nonce, $N_b \xleftarrow{R} \{0,1\}^n$, where n is 32 bits, and increments the sequence number by 1. B computes $Ciph = E_{K_{ab}}(id, seq, Nb)$ and $MAC(Ciph, K_{ab})$. B then sends to A $(Ack, Ciph, MAC(Ciph, K_{ab}))$, where *Ack* means acknowledgement.

Step 3: Upon receipt of $(Ack, Ciph, MAC(Ciph, K_{ab}))$, A checks the message integrity and computes $m = D_{K_{ab}}(Ciph)$. If all are successful then A knows that B has received the identity and disassociation token. A saves the random nonce, N_b.

Step 4: A chooses a random nonce $N_a \xleftarrow{R} \{0,1\}^n$ and a new sequence number $seq \xleftarrow{R} \{0,1\}^n$, where n is 32 bits. A sends the message $E_{+K_c}(Req, N_a, Challenge,$ $seq, \beta = g^y)$ together with his signature $SIG_A(Req, N_a, Challenge, seq, \beta = g^y)$, where β is A's Diffie-Hellman public component. Note that *Challenge* is some random bits that is sent by C in the *reserved* field of the Hello message (or we could add another field for holding the random bits to the Hello message).

Step 5: Upon the receipt of $E_{+K_c}(Req, N_a, Challenge, seq, \beta)$ and A's signature, C computes $m = D_{-Kc}(E_{+K_c}(Req, N_a, Challenge, seq, \beta))$ and verifies the signature. C checks his topology table to see if A has already been pre-authenticated, or if A is already a member of the network. If the signature is valid and A is already an existing member then A increments *seq* by 1 and sends to A, $E_{+K_a}(N_a, seq, \alpha = g^x)$ together with his signature $SIG_C(N_a, seq, \alpha = g^x)$, where α is C's Diffie-Hellman public component. C is now able to compute the pairwise session key $K_{ac} = \beta^x$. Here *Challenge* serves two purposes. Firstly, *Challenge* is returned to C so that C "knows" that the received packet is a "reply" to the Hello message that he sent earlier. Secondly, *Challenge* is used as a challenge that is required by the signature-based MT-authenticator. C erases x.

Step 6: Upon the receipt of $E_{+K_a}(N_a, seq, \alpha)$ and C's signature, A computes $m = D_{-K_a}(E_{+K_a}(N_a, seq, \alpha))$ and verifies the signature. If the signature is valid then A verifies the nonce N_a and the sequence number. If all are OK then A computes the session key $K_{ac} = \alpha^y$. A erases y. Both A and C have now established a new session key, K_{ac}.

Step 7: A holds the random token, *tok*, the sequence number, *seq* (the old *seq*), and the random nonce, *id*, from Step 0. A increments *seq* by 1. A computes $Ciph = E_{K_{ab}}(seq, id, tok)$ and $MAC(Ciph, K_{ab})$. A sends $(dis, Ciph, MAC(Ciph, K_{ab}))$ to C.

Step 8: Upon receipt of $(dis, Ciph, MAC(Ciph, K_{ab}))$, C looks at the destination address and forwards the message to B.

Step 9: Upon receipt of $(dis, Ciph, MAC(Ciph, K_{ab}))$, whose source address is A, B checks the message integrity and computes $m = D_{K_{ab}}(Ciph)$. If B's calculations are successful, B checks the sequence number. B then compares the received *id* and *tok* with the entries in his table. If the values match, then B disassociates A, indicating that A has now moved away and is no longer one of B's one-hop neighbours.

This section introduces a "correct" and secure handoff protocol, which solves the problem of micro-mobility management. The resultant protocol consists of five messages, which accomplish the handoff process, key establishment and the disassociation process.

Figure 4-2 summarises the handoff protocol. Note that the last message, the disassociation token, will be forwarded by the new node if the roaming node is already outside the coverage of the old node.

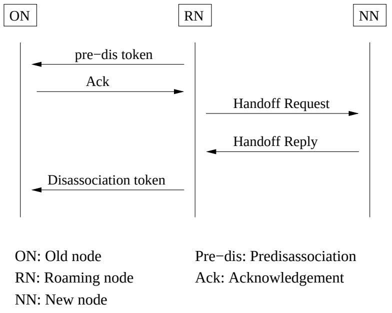


Figure 4-2: Summary of the handoff protocol

The reason that we have design the handoff protocol in such a way that A sends a disassociation token to B before initiating a handoff with C is because we would like B to be able to keep up-to-date the status of A, the roaming node, as it is about to move away.

Summary

This chapter gives the design of the authentication and secure handoff protocols that, we believe, are suited for pure mobile ad hoc networks. During the design, we clearly stated the problems that would need to be overcome, and the goals that would need to be achieved at the end of each protocol run. The protocols were then analysed using the combination of the BCK modular approach and the CK analysis method. The correctness of each of the protocols was proved using the GNY technique, during which we found that there was a problem with the original design of the secure handoff protocol. This led us to re-design a part of the protocol, re-analyse and then re-prove (the correctness of) the protocol. We presented the descriptions of the authentication and secure handoff protocols at the end of their section.

The authentication and handoff protocols are constructed by using a combination of well known cryptographic tools, namely RSA public-key cryptosystem and Diffie-Hellman key agreement. We have shown that only four messages are needed to complete the authentication and key establishment, while a handoff process can be accomplished in just five messages. The multipoint relay algorithm is integrated into the pre-handoff protocol in order to reduce the usage of the network bandwidth.

At first glance, one may wonder why we do not just do another authentication when roaming, since it only takes four messages to complete the authentication protocol rather than five messages for the handoff protocol. The main reason for it is that within the five messages of the handoff protocol we also achieve disassociation. In order to accomplish secure disassociation, three messages are needed. This means that it would take seven messages in total if we were to carry out the authentication protocol together with the disassociation process.

The authentication, pre-handoff and handoff protocols are closely related to one another. We can see that once an authentication is complete, there is a change in the network topology (since a new mobile node has just joined). The routing information and topology tables need to be updated. This is accomplished by the exchange of Hello messages among the mobile nodes. In the pre-handoff protocol, we extend this mechanism so that mobile nodes possess necessary information for future handoffs that may take place. During the pre-handoff stage, mobile nodes exchange security information with one another. This security information, namely the RSA public components, which is also the main components of the first message of the authentication protocol, is then used for the encryption and identification purposes while the handoff process is carried out.

The resultant protocols, we claim, are suitable and practical for local private mobile ad hoc networks. More importantly, they are correct (according to the GNY protocol) and secure in the unauthenticated-links model (according to the BCK and CK methods). The implementations and simulations of the protocols will be provided in the next chapter.

Chapter 5

Simulations and Implementations

In this chapter, we describe the simulations that we carried out in order to illustrate the working of our protocols - authentication and key establishment, pre-handoff, and handoff. These simulations are run using the network simulator *ns2* [MF]. The simulations are divided into three main categories with respect to the protocols designed in the previous chapter. At the end of this chapter, we will give an overview of how the protocols can be employed in a sample ad hoc network, and how the pre-handoff stage is implemented (authentication and handoff protocols are implemented according to the descriptions given in Chapter 4). Results of running the simulations are presented in the next chapter.

The simulations will range from a simple scenario with two or three mobile nodes, to a more complex ones with more than two hundred mobile nodes.

For the simulations, we use a Pentium IV 1.8GHz processor PC that has 256Mb of RAM and runs Redhat Linux 7.2. The network simulator is *ns2* version 2.27. The *Openssl* library is also applied for the protocol implementations. In ns2, we have followed an application-layer approach, and developed UDP-like transport agents that allow for message delivery of the actual protocols.

The simulations are carried out in two different settings or environments. Firstly, an outdoor environment, where the line of sight between two communicating parties (transmitter and receiver) is clear, and the condition is that the communication range is modeled as a circle around each mobile node. If a receiver is within the coverage, it receives all the packets. Otherwise, it loses all the packets. The maximum transmission range of any mobile node in this model is approximately ninety-five metres, according to our findings in the network simulator. This model is known as the *Two-ray ground reflection* model, which considers both the direct and ground-reflected propagation path between transmitter and receiver [New04]. More detail on this model can be found in [Pro03]. The following variables define the outdoor environment and the settings of the IEEE802.11b mobile nodes.

```
set val(chan)    Channel/WirelessChannel    ;# channel type
set val(prop)    Propagation/TwoRayGround   ;# radio-propagation model
set val(netif)   Phy/WirelessPhy            ;# network interface type
set val(mac)     Mac/802_11                 ;# MAC type
set val(ifq)     Queue/DropTail/PriQueue    ;# interface queue type
set val(ll)      LL                         ;# link layer type
set val(ant)     Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)  250                        ;# max packet in ifq
set val(rp)      DSDV                       ;# routing protocol

#Settings for IEEE802.11b
$val(mac) set SlotTime_          0.000020  ;# 20us
$val(mac) set SIFS_              0.000010  ;# 10us
$val(mac) set PreambleLength_    144       ;# 144 bit
$val(mac) set PLCPHeaderLength_  48        ;# 48 bits
$val(mac) set PLCPDataRate_      1.0e6     ;# 1Mbps
$val(mac) set dataRate_          11.0e6    ;# 11Mbps
$val(mac) set basicRate_         1.0e6     ;# 1Mbps
$val(netif) set freq_ 2.4e+9
$val(netif) set Pt_ 3.3962527e-2
```

The second environment is an office environment, where the line of sight between two communicating parties is obstructed, and the conditions are not ideal in that the coverage of each mobile node is not always a perfect circle. The maximum transmission range of any mobile node in this model is between five and seven metres. In this particular setting, each mobile node can be thought of as situating in a room surrounded by brick walls. This model is known as the

Shadowing model, and can be found in [Pro03]. The following variables define the office environment and the settings of the IEEE802.11b mobile nodes.

```
set val(chan)    Channel/WirelessChannel  ;# channel type
set val(prop)    Propagation/Shadowing    ;# radio-propagation model
set val(netif)   Phy/WirelessPhy          ;# network interface type
set val(mac)     Mac/802_11               ;# MAC type
set val(ifq)     Queue/DropTail/PriQueue  ;# interface queue type
set val(ll)      LL                       ;# link layer type
set val(ant)     Antenna/OmniAntenna      ;# antenna model
set val(ifqlen)  250                      ;# max packet in ifq
set val(rp)      DSDV                     ;# routing protocol

#Settings for IEEE802.11b
$val(mac) set SlotTime_          0.000020  ;# 20us
$val(mac) set SIFS_              0.000010  ;# 10us
$val(mac) set PreambleLength_    144       ;# 144 bit
$val(mac) set PLCPHeaderLength_  48        ;# 48 bits
$val(mac) set PLCPDataRate_      1.0e6     ;# 1Mbps
$val(mac) set dataRate_          11.0e6    ;# 11Mbps
$val(mac) set basicRate_         1.0e6     ;# 1Mbps
$val(netif) set freq_ 2.4e+9
$val(netif) set Pt_ 3.3962527e-2

#Set values of shadowing model
$val(prop) set pathlossExp_     4  ;# In building.  Obstructed.
$val(prop) set std_db_          7  ;# Office.  Hard partition.
```

From the above settings, the *path loss exponent* (pathlossExp_) is set to 4 which means that this is a building environment and the line of sight between any two mobile nodes is obstructed. The *shadowing deviation* (std_db_) is set to 7, which means that each mobile node is in an office separated from other mobile nodes by hard partitions. Figure 5-1 shows a pictorial representation of a sample setting in the indoor environment. It can be seen that mobile nodes are separated from one another by hard partitions. Note that the network simulator does not show the lines between mobile nodes. We have drawn them in Figure 5-1 because we would like to illustrate the separation of the nodes.

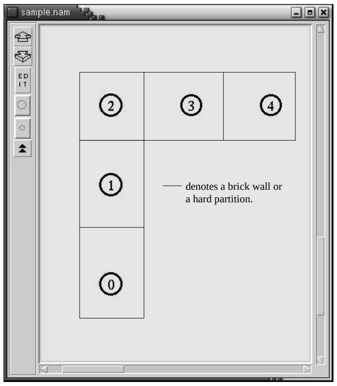


Figure 5-1: A typical indoor setting

The Tables 5.1 and 5.2 show the lengths of the messages involved in the authentication and handoff protocols respectively. We refer the readers to the descriptions of the respective protocols in the previous chapter for the reference of the message numbers.

Table 5.1: The lengths of authentication messages

Authentication message number	Message length (bytes)
1	139
2	389
3	388
4	260

Table 5.2: The lengths of handoff messages

Handoff message number	Message length (bytes)
1	168
2	136
3	272
4	264
5	172

The rest of the chapter describes the simulation scenarios for all three protocols - authentication, pre-handoff and handoff. Note that for all of the nodes in the simulations, there is background traffic among them. Specifically, there are ICMP messages being sent among the mobile nodes in all of the simulations while the authentication protocol, pre-handoff protocol and handoff protocol are being carried out.

5.1 Authentication and key establishment

The main objective when generating the simulations is to cover as many scenarios as possible. In this section, we have four different settings for the simulations. First, we let one pair of mobile nodes authenticate each other at any one time (one-to-one authentications). Secondly, more than one mobile node sends a request-to-join message to one node simultaneously (many-to-one authentications). Thirdly, one mobile node broadcasts a request packet, and the authentication and key establishment with all the nodes that receive the request packet is carried out (one-to-many authentications). Fourthly, a number of mobile nodes send a request packet to one another simultaneously to initiate authentication (many-to-many authentications).

5.1.1 One-to-one authentications (Scenario 1)

Figure 5-2 shows an example of how a mobile node could be positioned when trying to join an existing ad hoc network. Let's suppose that Node_(0) is an existing member of a private local mobile ad hoc network. Node_(1) is the first that would like to join. He sends a request packet to Node_(0) to start the authentication. Then Node_(2) and Node_(3) join the network afterwards, one at a time. Note that in this scenario, Node_(1), Node_(2) and Node_(3) are not within each other's range. That means that when a request is sent, only Node_(0) can receive it and, therefore, respond to it.

In addition, we place a pair of nodes randomly in such a way that they are within each other's radius, and let them authenticate one another. This is repeated ten times so that an average authentication time can be calculated.

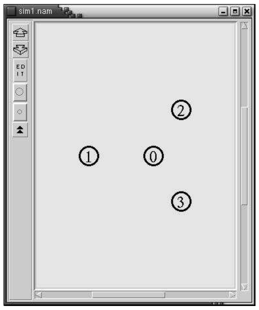


Figure 5-2: One-to-one authentications

5.1.2 Many-to-one authentications (Scenario 2)

The aim of this scenario is to test whether or not a mobile node can handle the situation where multiple request packets are sent to him simultaneously. Figure 5-3 shows an example of how the simulations are set up. In this particular setting, Node_(1), Node_(2), Node_(3) and Node_(4) all send a request-to-join packet to Node_(0) simultaneously. For these simulations, the number of mobile nodes that send the request packet will range from two to ten.

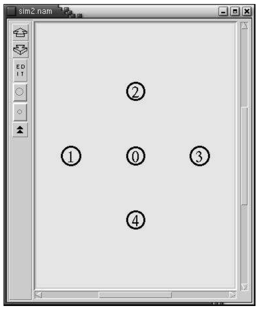


Figure 5-3: Many-to-one authentications

5.1.3 One-to-many authentications (Scenario 3)

In this scenario, Node_(0) broadcasts a request packet suggesting that he would like to join an existing ad hoc network. All of the existing members (up to ten nodes, in these simulation) who receive the packet, will respond accordingly. Node_(0) will, therefore, carry out the authentication and key establishment with all the reachable nodes. Note that the nodes around Node_(0) are positioned randomly each time the simulation is executed. Figure 5-4 depicts an example of the simulation setup.

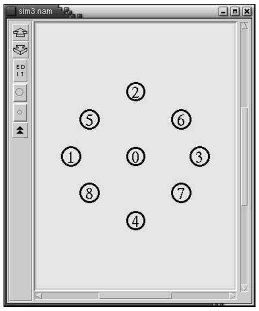


Figure 5-4: One-to-may authentications

5.1.4 Many-to-many authentications (Scenario 4)

In addition to the first three scenarios, we place a number of mobile nodes randomly in such a way that all of the nodes can communicate directly with one another. This simulation begins with one pair of nodes simultaneously sending a request packet to one another. The number of nodes is then increased up to twenty five nodes, all of which broadcast a request packet at the same time.

This scenario can be thought of as when mobile nodes are forming a private local mobile ad hoc network for the first time. Once the network is formed, any of the first three scenarios could happen when a new node or new nodes join the network.

The simulations, described above, are run in order to illustrate the authentication and key establishment protocol in a private local mobile ad hoc network. We begin the simulations with a simple scenario, where the authentication is between a pair of mobile nodes, and go up to a more complex ones. We have covered one-to-one authentications, many-to-one authentications, one-to-many authentications and many-to-many authentications.

After the authentication and key establishment, there is a change in the network topology. Therefore, the mobile nodes have to update their routing information as well as their topological information. The next section contains the description of how the simulations for the pre-handoff stage are done.

5.2 Pre-handoff

In this section, the simulations show the process of what happens after a mobile node or mobile nodes have been authenticated and have joined a private local mobile ad hoc network, i.e. when the network topology changes. At this stage, Hello messages are exchanged so that the necessary topological information can be updated. This includes the update of the list of the one-hop neighbours, the list of the two-hop neighbours and the topology table. Moreover, during this stage, a set of multipoint relays (MPRs) will be computed. The multipoint relays, after having been selected by each node, will send to that node the security information of the node's two-hop neighbours. The multipoint relay will also pass the security information of that mobile node to his two-hop neighbours. After receiving the necessary security information, the mobile node will compute a new set of Diffie-Hellman components (as stated in the description of the protocol in Section 4.2), then the mobile node will be ready to handle any handoff process that may occur.

Due to the fact that the performance and complexity of the Multipoint Relay algorithm is presented in [QVL00], and the exchange of the Hello messages and the updates of routing and topological information are already parts of most routing protocols (especially table-driven routing protocols), we will focus the simulations on the process of getting mobile nodes ready for future handoffs, which is the extension to the usual exchange of Hello messages. This process consists of the exchange of security information of the nodes' two-hop neighbours and the computation of Diffie-Hellman components, both of which will be necessary for the handoff protocol.

The purpose of running these simulations is that we would like to demonstrate the performance of this particular stage of the pre-handoff protocol. Each simulation will begin after all the nodes have been authenticated and have joined the network. The mobile nodes will then send a Hello message to inform each other of the up-to-date information regarding the topology of the network. Each mobile node will then process the packet according to the protocol description stated in Section 4.2. The simulation will end after the nodes have been informed of their two-hop neighbours' security information, and the Diffie-Hellman components have been computed, which will be needed for the future handoff process.

5.3 Bringing the two together

Having simulated and tested the authentication protocol in scenarios 1, 2, 3 and 4 as well as the pre-handoff stage, we are able to incorporate them into more complex settings, where there are a lot more mobile nodes involved. The main purpose of these simulations is that we would like to demonstrate that our protocols will work in larger settings as well as smaller and simpler ones. The simulations are carried out in the following network settings.

A number of mobile nodes are positioned randomly within an area of the size 800m x 800m. At the same time, all the nodes broadcast a request packet in order to initiate the authentication and key establishment protocol with the surrounding nodes. These scenarios can represent the situation where a number of mobile nodes are setting up a private local mobile ad hoc network for the first time (in a larger setting than that in Scenario 4 and not all of the mobile nodes are within each other's radius). Once the authentication is complete, the pre-handoff will begin. That is the nodes will start sending Hello messages, updating their topology information, selecting their multipoint relays, passing security information and computing Diffie-Hellman components.

We simulate the authentication protocol and the pre-handoff protocol with twenty mobile nodes, fifty nodes, one hundred nodes, one hundred and fifty nodes, two hundred nodes, and two hundred and fifty mobile nodes within the specified area. Note that we alter the number of nodes within the same area each time the simulation is run, because we would like to experiment the protocols with different node densities (i.e. *Number of nodes/area*).

The positions of the mobile nodes are generated randomly by a tool known as *setdest*, which is part of the network simulator, ns2. The protocol is run five times on each set of mobile nodes, where the positions of the nodes are different each time. This is done because we would like to vary the scenarios rather than restrict them to one particular setting. The diagrams of the network layouts can be seen in Appendix B.

Once the above simulations have finished, i.e. the network has been set up, any of the authentication scenarios can take place. We simulate this by "adding" mobile node or mobile nodes to the existing network, and let them carry out the authentication process as usual. Once the node or nodes have been added as new members of the network, the pre-handoff protocol will again be invoked (as there is a change in network topology).

The simulations in this section will allow us to learn whether or not our authentication and pre-handoff protocols will be able to cope with larger and more complex network settings.

5.4 Handoff

By completing the authentication and pre-handoff protocols, mobile nodes now hold enough necessary information to carry out the handoff protocol. In this section, we provide the simulations for single handoffs in the outdoor and indoor environments, simultaneous (multiple) handoffs and handoffs with non-stationary mobile nodes.

We have mentioned at the end of the previous chapter that at first glance, it may appear simpler to just do another full authentication when roaming to a new position. However, this is not the case, since the four messages needed by the authentication protocol do not include secure disassociation. In order to achieve secure disassociation with the mobile node that the roaming node is moving away from, three more messages are needed. This means that it would take seven messages to complete a handoff if the authentication protocol were used as part of the process, whereas the handoff protocol, designed in Section 4.3, takes just five messages. This is the main reason why we have decided to apply the handoff protocol rather than doing the re-authentication.

5.4.1 Outdoor environment (single handoffs)

The handoff protocol is illustrated by having a mobile node roam from one position to another in such a way that the node leaves the coverage of a node and enters that of another. The speed of the movement will be varied from the average walking speed ($1.34ms^{-1}$) to the average speed of Eurostar ($83ms^{-1}$), which are as follows.

- The average walking speed, $1.34ms^{-1}$
- The average running/jogging speed, $5.00ms^{-1}$

- The speed of a car travelling at 30 mph, $13.4ms^{-1}$
- The speed of a car travelling at 60 mph, $26.8ms^{-1}$
- The average speed of Eurostar, $83.0ms^{-1}$

The simulations are carried out in this way in order to demonstrate that the transfer of the necessary data, the handoff process itself and the disassociation process can be executed and completed quickly enough for the nodes to roam without losing the connection. We would also like to illustrate that the non-moving nodes are also capable of handling the handoff process. Figure 5-5 shows how the simulation is set up.

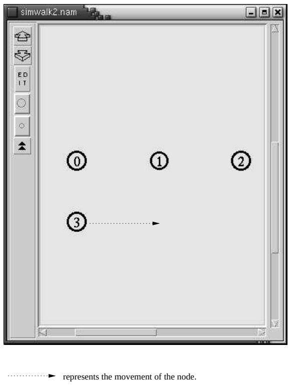


Figure 5-5: Handoff simulation set up

Node_(0) and Node_(1) are neighbours. Node_(1) and Node_(2) are each other's neighbour, and Node_(3) is the other one-hop neighbour of Node_(0)'s. Here, Node_(3) is the roaming node, and will move from Node_(0) towards Node_(2). Node_(3) is currently within the coverage of Node_(0) only. As he approaches Node_(1), the handoff process begins. The same process is repeated when Node_(3) approaches Node_(2).

All of the nodes in all of the simulations have already been authenticated, and therefore are members of the network. Before the handoff protocol begins, all the nodes will have exchanged and computed all the information necessary for the handling of the handoff process (i.e. pre-handoff will have already been done prior to this).

In addition, after the handoff protocol is finished, the pre-handoff process will be carried out. That is Hello messages are exchanged due to the change in network topology, and topological information are updated. As a result of the pre-handoff phase, the mobile nodes will again be ready to handle any future handoffs that may take place. This means that it is important for the mobile nodes, especially the travelling node, to complete this process before a new handoff process begins. The following simulations (using the setup shown in Figure 5-5) will determine how many neighbours Node_(1) can have, so that when Node_(3) finishes the handoff protocol with Node_(1) (Node_(1)'s neighbours will become Node_(3)'s two-hop neighbours), he will have enough time to complete the pre-handoff phase before initiating the handoff with any of Node_(1)'s neighbours. In other words, we would like to find out the maximum number of two-hop neighbours that Node_(3) can have, so that he can complete the pre-handoff stage before the next handoff can begin with any of those two-hop neighbours.

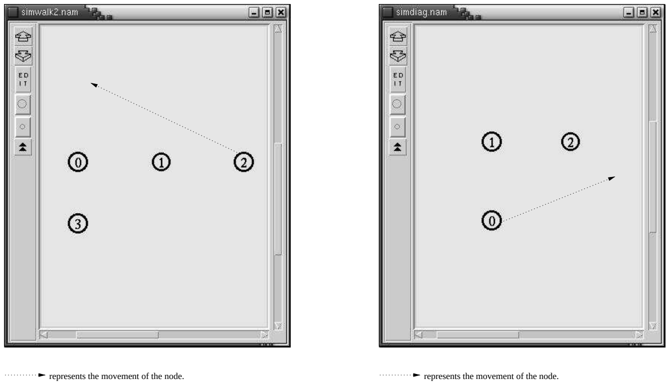
The simulation setups will be the same as before. The speed of the travelling mobile node will also be varied. For each speed, we will keep adding a new one-hop neighbour to Node_(1) until the roaming node, Node_(3), cannot complete the pre-handoff stage before the next handoff process begins with Node_(2).

5.4.2 Indoor environment (single handoffs)

There are three scenarios which are considered in the office environment. First of all, the setup is similar to the one shown in Figure 5-5, where the roaming node, Node_(3), moves in a straight line from Node_(0) to Node_(2) (which could represent a node moving in a straight line along a corridor). Secondly, the roaming node moves diagonally from one node to another (which could represent a node moving past a corner inside a building). Figure 5-6 depicts a couple of examples of this scenario.

Special case of handoff

The third scenario or the special case is as follows. Figure 5-7 shows that Node_(3) is roaming towards Node_(0). When Node_(3) approaches Node_(0), it is expected that Node_(3) and Node_(1) (which is Node_(3)'s two-hop neighbour) carry out the handoff mechanism. However, at that point, it is likely that the signal from


simwalk2.nam
ED IT
0
1
2
3
represents the movement of the node.
simdiag.nam
ED IT
1
2
0
represents the movement of the node.

(a) First diagonal movement

(b) Second diagonal movement

Figure 5-6: the roaming node moves diagonally.

Node_(0) will actually be stronger than the signal from Node_(1) (Node_(3) is closer to Node_(0) than Node_(1)). Therefore, Node_(3), by the definition of the handoff process, should carry out the handoff process with Node_(0) rather than Node_(1). The problem is that Node_(3) does not hold any security information of Node_(0) since it has not been one of Node_(3)'s one-hop or two-hop neighbours. This means that before the handoff process can commence, Node_(3) will have to contact Node_(2) in order to obtain the security information that belongs to Node_(0). Node_(0) will have to contact Node_(1) so that he can obtain Node_(3)'s security information. Note that Node_(2) holds the information of Node_(0) and Node_(1) holds the information of Node_(3) due to the pre-handoff process that has been done prior to this stage. Once these exchanges of messages is finished, Node_(3) and Node_(0) can continue processing the usual handoff protocol.

Alternatively, for the special case of the handoff protocol, re-authentication could be carried out by the roaming mobile node and the node it is approaching. This process works as follows. With reference to Figure 5-7, as Node_(3) moves away from Node_(2), according to the description of our handoff protocol, Node_(3) sends a disassociation token to Node_(2), who replies with an acknowledgement. When Node_(3) approaches Node_(0), they will carry out the full authentication protocol. That is Node_(3) starts the protocol by sending a request

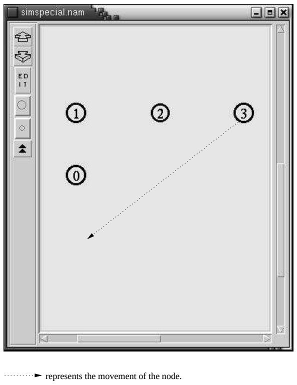


Figure 5-7: Handoff simulation (special case)

message to Node_(0). Mutual authentication and key establishment between the two mobile nodes will then be completed in the subsequent packets in accordance with the description of the authentication protocol in Section 4.1.4. Once that is finished, Node_(3) will send a disassociation message to Node_(2) in order to end the handoff process.

5.4.3 Simultaneous handoffs

All of the previous handoff simulations are concerned with single handoffs. That is mobile nodes handle one handoff at any one time. Now, we would like to illustrate whether or not our handoff protocol will be able to manage simultaneous handoffs as well. For these simulations, we let mobile nodes roam from Node_(1) to either Node_(0) or Node_(2). At any one time, Node_(1) will have to handle more than one disassociation_wait packet and more than one disassociation packet. Node_(0) or Node_(2) will have to handle more than one handoff_request packet or one handoff process at once. By running these simulations, we should be able to show whether or not simultaneous handoffs are possible.

Figure 5-8 shows an example of how these simulations are set up.

5.4.4 Handoffs with non-stationary nodes

So far we have only covered the situation where a travelling mobile node leaves the coverage of a non-moving mobile node and enters the coverage of another stationary mobile node. Here, we present the simulations for the situation, where

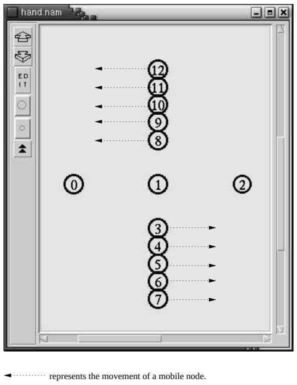


Figure 5-8: Simultaneous handoffs

a mobile node carries out the handoff process with another mobile node, which is also moving, but in a different direction.

Figure 5-9 shows an example of how the simulations are set up in this section. From the diagram, the network is laid out as follows. Node_(0) and Node_(2) are one-hop neighbours of Node_(1). Node_(1), Node_(3) and Node_(4) are within each other's radius. Similarly, Node_(1), Node_(5) and Node_(6) are each other's one-hop neighbours. Moreover, Node_(0) and Node_(2) are nodes 3, 4, 5 and 6's two-hop neighbours.

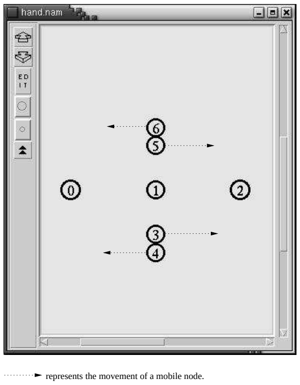


Figure 5-9: Handoff with non-stationary nodes

For this particular simulation, we let Node_(3) roam to Node_(2) while Node_(4) moves to Node_(0). At the same time, we let Node_(5) roam to Node_(2) and Node_(6) to Node_(0). All of the node movements happen simultaneously.

What will we learn from these simulations? What we have seen so far, as explained earlier, is the situation, where one mobile node roams away from a stationary mobile node. These simulations will show whether or not our handoff protocol can cope with the situation where a mobile node moves away from a stationary mobile node as well as away from a mobile node that is moving in a different direction. With reference to Figure 5-9, it can be seen that nodes 3 and 4 are moving away from each other in opposite directions, while they are both roaming away from Node_(1), which is stationary.

In the above sections, we have explained the different scenarios that are simulated in order to illustrate the working of our protocols - authentication and key establishment, pre-handoff and handoff. We set up the simulations in two extreme environments. They are the outdoor environment (where there is no obstruction between two communicating nodes) and the non-ideal office environment (where the path between two mobile nodes is blocked by hard partitions). The results of these simulations are provided in the next chapter. The next section explains how the three protocols - authentication, pre-handoff and handoff - fit together and how they are implemented.

5.5 Protocol overviews and implementation

This section will describe the authentication protocol, pre-handoff and handoff mechanism, starting from the moment node X joins to when the handoff process is complete. The protocols are implemented in accordance with the protocol descriptions and specifications explained in Chapter 4.

We now describe how our protocols can be employed in a sample ad hoc network. Let's say that there is an existing ad hoc network, which is shown in Figure 5-10. Figure 5-10 displays the layout of the sample mobile ad hoc network. The lines connected two mobile nodes denote that the two nodes are neighbours. For example, from the diagram we can see that Node_(A) is connected to nodes F and G. Therefore, Node_(A) is a one-hop neighbour of both nodes F and G. Another example is that Node_(C) is connected to nodes H, I and L. Therefore, Node_(C) is a one-hop neighbours of nodes H, I and L.

X joins the network by carrying out the authentication and pairwise key

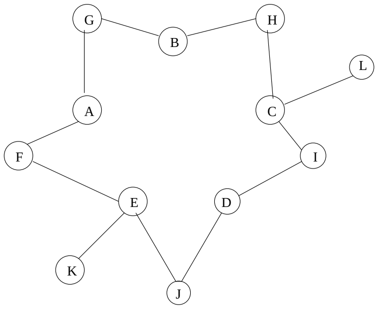


Figure 5-10: A sample mobile ad hoc network

establishment. The process begins by X broadcasting his Request-to-Join packet. A, B, C, D and E reply and the rest of the protocol is executed. This is basically the same as Scenario 3 (see Section 5.1.3), i.e. X authenticates and establishes pairwise session keys with all of the reachable nodes - A, B, C, D and E in this case. Having established a pairwise session key with A, B, C, D and E, X is now part of the network and a new one-hop neighbours of nodes A, B, C, D and E. Figure 5-11 depicts the network after X has joined.

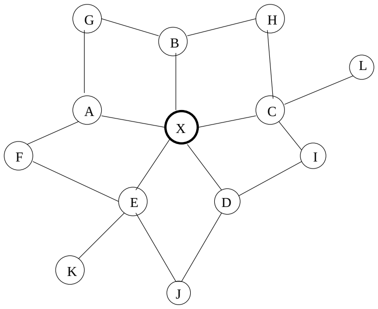


Figure 5-11: Node_(X) has joined the mobile ad hoc network.

X now has a list of his one-hop neighbours - A, B, C, D and E. The list is created and updated as X executes the authentication and key establishment protocol. Since there is a change of network topology, i.e. X has just joined the wireless network, the next step is the pre-handoff protocol. That is the mobile nodes will begin sending each other *Hello* messages, so that they get the up-to-date topological information. Each of X's neighbours will also send him a Hello message containing a list of his one-hop neighbours. Once the Hello message is received, X updates his topology table and create a list of the two-hop neighbours.

Table 5.3: Node_(X)'s topology table

A	F	G	
B	G	H	
C	H	I	L
D	I	J	
E	F	J	K

Table 5.4: Node_(X)'s list of two-hop neighbours

F	2
G	2
H	2
I	2
J	2
K	1
L	1

X's topology table, created by the program implemented, is shown in Table 5.3.

Note that the first column contains the one-hop neighbours of X. Other columns are the one-hop neighbours of the node in the first column. For example, F and G are the neighbours of A. One might ask why X is not in the table since it is also a neighbour of all the nodes in the first column. We have implemented the function to create the topology table in such a way that the host itself, X in this case, is not added to the list.

X's list of two-hop neighbours will then be generated and will look like Table 5.4. This has to be a table rather than a list, because there needs to be one extra column which indicates the number of neighbours that each of the two-hop neighbours of X has. This information will be useful for later algorithm, namely the Multipoint Relay algorithm.

The first column is the list of X's two-hop neighbours. The second column, as mentioned before, is the number of neighbours that the node in the first column has.

Before the roaming process can take place, X wants to receive the "security information" of all of his two-hop neighbours as well as pass his own information to the two-hop neighbours. Instead of asking all of his one-hop neighbours to pass the "security information" to X and X's two-hop neighbours, X selects a set of nodes to relay the message. This is done in order to reduce the number of packet duplications and to reduce the traffic in the wireless network. The algorithm that will be used to select the replaying nodes is called the *Multipoint Relay* algorithm. The nodes that are selected will be called *Multipoint Relays* or MPRs. The formal algorithm is explained in Section 4.2.

Next how the MPR algorithm is coded will be explained. The topology table and the two-hop table have been created by the earlier processes, Steps 1 and 2 of the algorithm are now ready to be implemented.

First, we look at the two-hop table to find a node, whose second column contains 1, i.e. has one neighbour. Once found, we compare the node to the list of nodes in the topology table. When a match is found, we add the node in the first column of that row (in the topology table) to the multipoint relay set. Then another node (in the two-hop table), whose second column contains 1 is looked for. This step is repeated until all the nodes with 1 in the second column are found. At the end of this step, we will end up with an initial list of multipoint relays. Next, Step 3 of the MPR algorithm will be implemented.

We create a list of nodes that are covered (can be reached from X via the multipoint relays) by the selected MPRs. This is done by working with the topology table and the list of MPRs. We look at the first item in the MPR list and compare that with the nodes in the first column of the topology table. If the match is found, the nodes in that row of the table (excluding the one in the first column) are added to the list of covered nodes. Then look at the next item in list of MPRs, and again try to find a match in the topology table. When found, instead of simply adding all the nodes in that row to the list of the covered nodes, we will have to check whether they are already in the list or not. If a node already exists in the list, it is discarded. If not, it is inserted in the list.

Now the list of the nodes that are covered by the multipoint relays has been created. The next thing we will do is create a list of the nodes that have not been covered by the MPRs, i.e. the remaining two-hop neighbours. This step can be done by subtracting the list of the covered nodes from the first column of the two-hop table. The difference between the two sets will be the set of the uncovered nodes. Once this process is done, we will be able to know which nodes cannot be reached by the existing set of the multipoint relays.

In order to select more MPRs (to cover the remaining two-hop neighbours), another table is created, which will show the number of nodes that are not yet covered by the current set of the multipoint relays. For this table to be generated, we use the existing tables including the topology table, the list of the multipoint relays and the list of the uncovered nodes. First, we take a node from the first column of the topology table and compare it with the existing MPRs. If the node is not already a multipoint relay, we will add it to the second column of this new table. In the same row as the node we have chosen, we compare all the nodes to the list of the uncovered nodes and count the number of matches. We then add the number of matches to the first column of the new table, in the same row as before. By looking at the table, we can tell how many nodes, not yet covered, a non-MPR node has. This process is carried out until all the non-MPRs are selected. The table is sorted in the order of the number of matches. Note that if the number of matches is zero, we do not add that node to the table.

Selecting a multipoint relay is now an easy task. We will just have to look at the newly created table and pick the first node in the table. This is because that node has the maximum number of nodes uncovered by the current multipoint relays. Once the selected node is added to the MPR list, we repeat the whole process. This is done until the list of the uncovered nodes is empty, i.e. all of the two-hop neighbours are covered. At the end of the algorithm, we will have obtained a list of multipoint relays. In this example, the multipoint relays are nodes B, C and E. Figure 5-12 shows that nodes B, C and E are the multipoint relays of Node_(X).

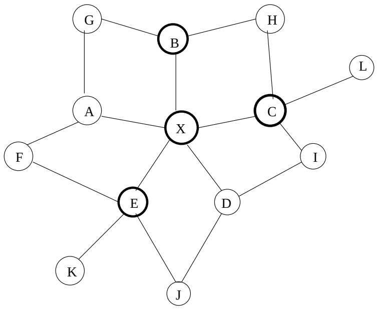


Figure 5-12: Multipoint relays: Node_(B), Node_(C) and Node_(E)

Now that X has selected his multipoint relay set, he will need to ask them to pass his “security information” to his two-hop neighbours as well as ask them to send him the public key components of the two-hop neighbours. Remember that when joining the network, X broadcast a request packet, including his public key components, to all of his neighbours, A, B, C, D and E. That means that the MPRs, B, C and E are already holding X’s public key components and those of X’s two-hop neighbours. We can see that by asking the nodes B, C and E to pass/relay the messages, all of X’s two-hop neighbours will be able to receive the message. These public key components are needed for the identification process when X roams into the range of any of the two-hop neighbours. The arrows in Figure 5-13 denote how the messages are relayed to and from X.

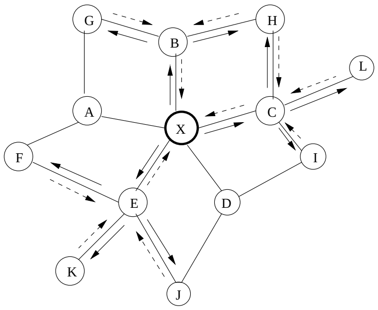


Figure 5-13: Multipoint relays passing information

For a handoff process, let’s say that X roams from where he is to node L. The handoff protocol will be processed in accordance with the protocol description (see Section 4.3.6 for detail). The pre-handoff process can then be repeated in order to get the up-to-date topological information to the mobile nodes, and to get X and other mobile nodes ready for the next handoff.

Summary

This chapter describes and explains how the protocols - authentication, pre-handoff and handoff - are simulated and tested. Different simulations ranging from simple to more complex are set up in order to cover as many scenarios as possible. The simulations are set in two different environments, outdoor and indoor.

For the authentication protocol, we have covered one-to-one authentications, one-to-many authentications, many-to-one authentications and many-to-many authentications. For the pre-handoff protocol, we will find out (in the next chapter) how long it takes to compute or process security information that is passed among mobile nodes. For the handoff protocol, we have set up the simulations to ensure that we have covered both single and multiple handoffs as well as handoffs with stationary and non-stationary nodes. The special cases of handoff, where the roaming node does not hold any security information of the node he or she is approaching, will also be simulated. These simulations include the re-authentication method as well as the normal handoff protocol with additional messages.

At the end of the chapter, we explain how the three protocols can be employed in a private local mobile ad hoc network, and how they are implemented.

Next chapter provides the results of the simulations and the evaluations of all of the protocols - authentication, pre-handoff and handoff.

Chapter 6

Results and Evaluations

Each section in this chapter is divided into two main parts - results and evaluation. First of all, the results sections will consist of the results that are obtained from the simulations described in Chapter 5. The results presented here include the time taken to complete the initialisation stage, the time taken to complete the authentication protocol (which includes both the packet processing time and the transmission time), the time taken to process security information in the pre-handoff stage and the time taken to complete the handoff protocol (which also includes both the processing time and the transmission time). We will present the results in the order of the simulations run in the previous chapter.

Note that in this chapter, what we mean by “outdoor” is that the line of sight between two mobile nodes is clear. When we say “indoor”, however, we mean that each mobile node is situated in a room, hence the line of sight between two mobile nodes are obstructed by hard partitions.

The results will then be analysed and discussed in the evaluation parts. We will discuss the results in order to see whether or not the protocols meet the criteria set in Chapter 4, and whether or not the protocols can overcome the problems that the existing protocols have (these problems are stated in Chapter 3).

The first two sections will present the results and evaluations for the initialisation stage and the authentication and key establishment protocol. They will be followed by the results and evaluations for the pre-handoff and handoff protocols.

It should be reiterated that all of the simulations are run on a Pentium IV 1.8GHz processor PC that has 256Mb of RAM and runs Redhat Linux 7.2. The network simulator is *ns2* [MF] version 2.27. The *Openssl* library is also applied for the protocol implementations.

Note that we validate the simulations and protocols by monitoring the outputs and activities of all of the mobile nodes in the simulated networks. This way, we can be sure that the mobile nodes and the protocols behave in the ways that we expect.

6.1 Initialisation

We refer back to Step 0 in Section 4.1.4. We can see that Step 0 is the initialisation stage, i.e. computing the RSA and Diffie-Hellman components prior to initiating the authentication. In this section, we will present the time it takes to compute the RSA and Diffie-Hellman components. We run the calculations of RSA and Diffie-Hellman separately. The times recorded in the table are the average times taken to complete each calculation. Each function is looped fifty times. The total time is then divided by fifty in order to give an average of an individual time. This process is repeated ten times. The results obtained from this experiment are shown in the Table 6.1.

6.1.1 Results

Table 6.1 shows how long it takes to compute the RSA and DH components. The times in the table have been rounded to three significant figures. Figure 6-1 is the bar chart generated from Table 6.1. The bar chart shows the time taken to complete the initialisation stage each time the protocol is executed. From the values we have in the table, on average it takes 3.09×10^{-5} seconds to complete the computation of RSA components, and 5.67×10^{-5} seconds to compute the Diffie-Hellman components. Overall, the mean value of the time taken to complete the initialisation step is 8.76×10^{-5} seconds.

6.1.2 Evaluation

The RSA and Diffie-Hellman public and private components are calculated at this stage so that the time taken to complete the actual authentication and key establishment protocol can be reduced. The Diffie-Hellman calculation could

Table 6.1: Time taken to compute RSA and DH components

	RSA ($\times 10^{-5} seconds$)	DH ($\times 10^{-5} seconds$)
1	3.28	5.43
2	7.73	6.45
3	2.34	7.66
4	2.85	4.69
5	2.85	6.76
6	1.95	4.65
7	2.11	6.59
8	2.62	4.26
9	3.01	5.55
10	2.15	4.68

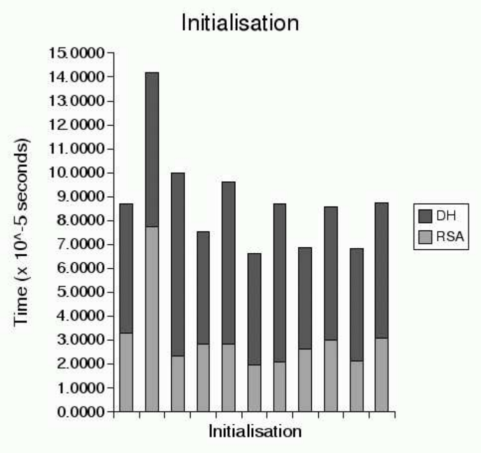


Figure 6-1: Time taken to complete the initialisation stage

actually be integrated into the authentication protocol itself. However, if that were the case, there could be a short delay during the authentication (hence it would take longer), which is not what we want.

The results suggest that on average it takes approximately 0.000031 seconds in order to complete the initialisation step. Every mobile node will have to complete this initialisation before initiating the authentication protocol. Even though 0.000031 seconds may not seem to be a noticeable delay, it is a good idea to pre-compute these components prior to the actual authentication, so that the time taken to carry out the authentication protocol can be shortened.

6.2 Authentication and key establishment

For each scenario in Section 5.1, the results of the authentication protocol are categorised into the results for outdoor environment and indoor environment. We first present the time taken for a pair of mobile nodes to authenticate each other. This will be followed by the results for many-to-one authentications, one-to-many authentications and many-to-many authentications. Note that the times recorded in the tables are the times obtained from the ns2 simulation files (or trace files).

6.2.1 One-to-one authentications (Scenario 1)

This scenario is where only two mobile nodes authenticate each other at any one time. First of all, the outdoor environment, Table 6.2 shows the time taken to complete the authentication process each time the simulation is run. From the results we have gathered, the average time taken for the authentication protocol to complete between two parties is 0.0099 seconds or 9.90 milliseconds.

Secondly, the indoor/office environment, the times recorded when running the simulation are shown in Table 6.2. On average in the indoor/office environment, the time taken to complete the authentication and key establishment between two mobile nodes is 0.0101 seconds or 10.10 milliseconds. These times exclude the initialisation step.

Table 6.2: Time taken to complete the authentication (in seconds)

Authentication number	Outdoor	Indoor
1	0.0094	0.0094
2	0.0094	0.0094
3	0.0094	0.0094
4	0.0094	0.0094
5	0.0090	0.0094
6	0.0090	0.0094
7	0.0090	0.0101
8	0.0111	0.0116
9	0.0110	0.0110
10	0.0120	0.0120

6.2.2 Many-to-one authentications (Scenario 2)

In scenario 2, more than one mobile node send a request message to the same existing member simultaneously. Table 6.3 records the results that we have gathered from running the simulations in outdoor environment and indoor environment. The results are plotted in the graph shown in Figure 6-2.

Table 6.3: Time taken to complete the authentication when receiving multiple request packets

Number of nodes	Time outdoor (s)	Time indoor (s)
2	0.0192	0.0212
3	0.0267	0.0279
4	0.0384	0.0397
5	0.0477	0.0494
6	0.0554	0.0569
7	0.0680	0.0692
8	0.0782	0.0806
9	0.0877	0.0912
10	0.0937	0.0958

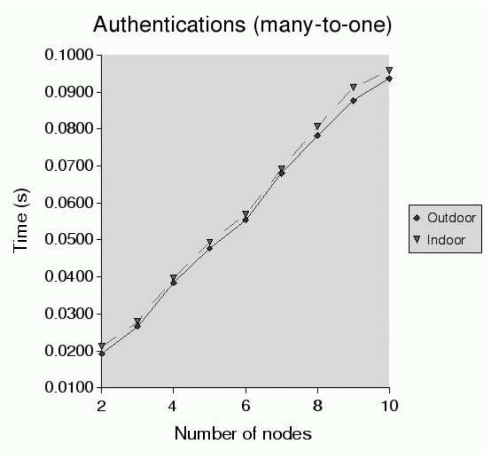


Figure 6-2: Time taken to complete the authentication when receiving multiple request packet

6.2.3 One-to-many authentications (Scenario 3)

This scenario is where a mobile node broadcasts a request packet, and those that receive the packet will respond accordingly, i.e. the joining mobile node simultaneously carries out the authentication with two or more mobile nodes. The simulations are run with two up to ten neighbouring nodes (i.e. the request message reaches two up to ten mobile nodes), and the times it takes to complete the process are recorded. Again, we run the simulations in both outdoor and indoor environments. The Table 6.4 shows the results that we have obtained. The results are plotted and the graph is shown in Figure 6-3. Note that each of the times in the Table 6.4 is an average time taken to complete the authentication with n nodes.

Table 6.4: Time taken to complete the authentication with n nodes

Number of nodes	Time outdoor (s)	Time indoor (s)
2	0.0236	0.0256
3	0.0298	0.0312
4	0.0379	0.0365
5	0.0451	0.0456
6	0.0526	0.0506
7	0.0595	0.0647
8	0.0626	0.0676
9	0.0710	0.0767
10	0.0802	0.0806

From Figure 6-3, it can be seen that the relationship between the time taken to complete the authentication and the number of mobile nodes is approximately linear. As a result, we can form one equation for the outdoor environment and one equation for the indoor environment (even though the times recorded for both environments are almost identical), so that it will be possible to estimate the time it would take one mobile node to carry out the authentications with n mobile nodes. Equation 6.1 is a linear equation for the outdoor environment and Equation 6.2 is a linear equation for the indoor environment. For both equations, y is the time and x is the number of mobile nodes.

$$y = 0.007075x + 0.00945, \tag{6.1}$$

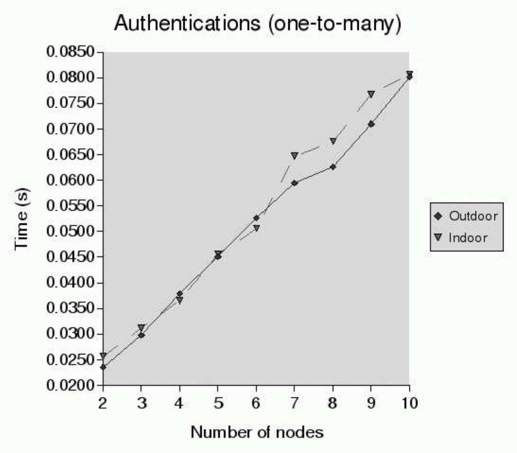


Figure 6-3: Time taken to complete the authentication protocol

where $x \geq 2$. Equation 6.1 approximately has the sum of squared absolute error of $1.358e-05$.

$$y = 0.006875x + 0.01185, \tag{6.2}$$

where $x \geq 2$. Equation 6.2 approximately has the sum of squared absolute error of $3.729e-05$.

6.2.4 Many-to-many authentications (Scenario 4)

This is the scenario where a number of mobile nodes are randomly positioned, and the authentication protocol is carried out by all the nodes as if they are setting up a network for the first time. In this simulation, all the nodes are placed in such a way that all of them are within each other's coverage.

Again we have obtained the results, i.e. the time taken to complete all of the authentications, for the outdoor and indoor environments. The times in Table 6.5 are the average times calculated for each set of mobile nodes. Figure 6-4 represents the results for the outdoor setting and indoor setting.

From the graphs shown in Figure 6-4, we can see that there is a non-linear relationship between the number of mobile nodes and the time taken to form a network (or to authenticate one another). Specifically, an equation, which can be used to estimate how long it would take n mobile nodes to set up a private

Table 6.5: Forming a mobile ad hoc network

Number of nodes	Time outdoor (s)	Time indoor (s)
2	0.0202	0.0202
3	0.0378	0.0378
4	0.0977	0.0970
5	0.1289	0.1370
6	0.1738	0.1744
7	0.2703	0.2721
8	0.3269	0.3288
9	0.4668	0.4658
10	0.5893	0.5907
15	1.1634	1.1659
20	2.0322	2.0521
25	3.8425	3.8526

mobile ad hoc network, has been formed. Equation 6.3 and Figure 6-4 show that there is approximately a quadratic relationship between the time and number of mobile nodes. In the equation, y represents the time taken to "form" an ad hoc network, and x is the number of mobile nodes. Remember that in this scenario, all of the nodes are within each other's range.

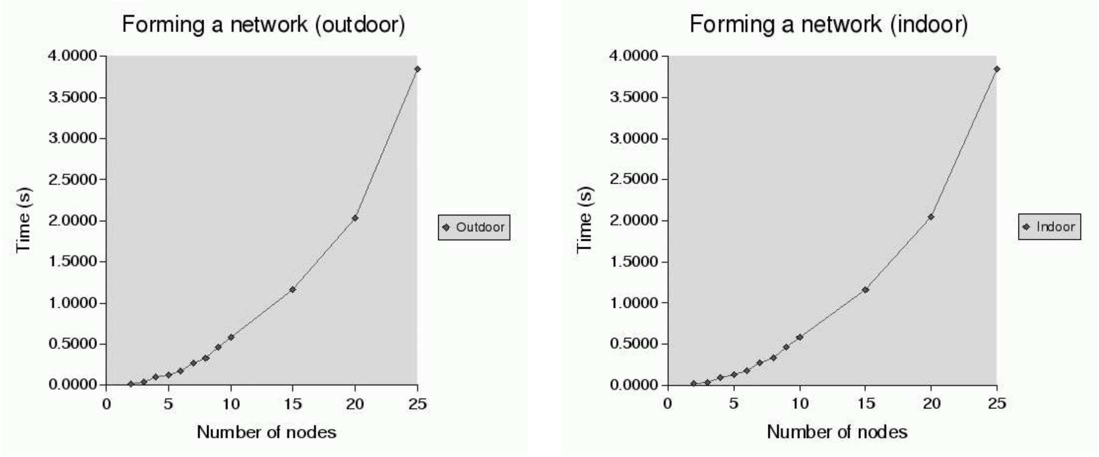
$$y = 0.00505x^2, \tag{6.3}$$

where $x \geq 2$. Equation 6.3 approximately has the sum of squared absolute error of 0.1537.

6.2.5 Evaluation

The main purpose for simulating the four scenarios is that we would like to ensure that the authentication protocol will work in all of the possible situations. They include one-to-one authentication (scenario 1), many-to-one authentication (scenario 2), one-to-many authentication (scenario 3) and many-to-many authentication (scenario 4).

Once the initialisation is done, the authentication protocol can be carried out. The first scenario shows that the time taken to complete the authentication and key establishment between two parties is approximately 10 milliseconds in both



(a) Forming a network outdoor

(b) Forming a network indoor

Figure 6-4: Time taken to form a network when all mobile nodes are in each other's coverage

outdoor and indoor environments.

In the second scenario where a mobile node receives more than one request packet (many-to-one), the results and graph show that the time taken that node to complete the authentication protocol with all other nodes is approximately directly proportional to the number of the request packets he receives. It is not surprising to see that the time taken to complete the authentication in this scenario is approximately n × *the time taken to complete the authentication with one node*, where n is the number of received request packets. This is because a full authentication and key establishment has to be done with each of the mobile nodes.

The graph of scenario 3, where a mobile node broadcasts a request packet, shows an approximately linear relationship between the time and the number of nodes that the request packet can reach. Some of the times recorded appears to be a little bit longer than n × *the time taken to carry out authentication with one node*, where n is the number of reachable mobile nodes. The reason is that if and when the broadcasting node receives more than one reply packet, it is necessary to compute a new set of Diffie-Hellman components for each of the additional replies. This is because it is important to have a different set of private and public components for establishing a different pairwise session key. If the same set of

Diffie-Hellman components were used for all key establishments, perfect forward secrecy and backward secrecy would not be achieved. In other words, each of the keys established would not be entirely unique, which would make it more vulnerable to attacks by passive adversaries (who know a subset of the key). By computing a new set of Diffie-Hellman components, we guarantee backward secrecy and prevent known-key attacks. That is, by using the knowledge of the old keys, the adversaries will not be able to establish the new keys. A full description of the known-key attacks can be found in Section 12.17 of [MvOV96]. Furthermore, this can prevent any passive adversaries from discovering any preceding keys by using the knowledge they have on the newly established keys. Hence, perfect forward secrecy is achieved. A formal description of forward secrecy can be found in Section 12.16 of [MvOV96].

The results and graph obtained from running the simulations in scenario 4 show how long it takes mobile nodes to complete the authentication and key establishment when all of the nodes initiate the protocol by broadcasting a request packet simultaneously. The times recorded in the table suggest that it does take longer than *n × the time taken to carry out authentication with one node.* This is because each mobile node sends a request packet and also receives a request packet, which means that potentially the authentication protocol will be carried out twice by each pair of the nodes. However, the results in Table 6.5 indicate that it does take less than *2 × n × the time taken to carry out authentication with one node.* This indeed is what we expect since the protocol is implemented in such a way that when this situation occurs (i.e. a node sends a request and also receives a request), the established keys are compared with one another. That is if they are already shared between two parties, there will not be any need to complete the other outstanding authentication and key establishment.

We have covered all of the possible situations that could take place when joining a private local mobile ad hoc network. The results have been obtained. Now we will discuss whether the performance of the authentication and key establishment protocol meets the criteria set in Section 4.1.1, and whether it can overcome the problems (Section 3.1) that the existing authentication protocols have.

Section 4.1.1 states the three goals that the authentication and key establishment protocol must achieve. They are mutual authentication, pairwise key establishment and protocol efficiency.

Firstly, by completing the authentication protocol, mutual authentication is automatically achieved. Mutual authentication is achieved within the first three messages of the protocol in that both parties will have proved to one another that they hold the same shared secret, K, hence they are authorised to be members of the same ad hoc network. In more detail, the request message is encrypted with the shared secret, K. If the recipient (of the request packet) also possesses the same secret, he will be able to decrypt the message. Some of the decrypted information is then used as part of the reply packet, specifically the sequence number and the initiator's random nonce. In addition, the reply packet contains a challenge text. When the initiator receives the reply packet, he checks whether the nonce and the sequence number match his own nonce and the expected sequence number respectively. If they match, he knows that the party that sent the reply message must have the same shared secret, K. The received challenge text is then encrypted using the secret, K. The encrypted challenge is sent back to the other node. Upon the receipt of the encrypted challenge, the receiver encrypts his challenge that was sent earlier. Both encrypted challenge texts are compared. If they are the same, then both parties must possess the same shared secret, K. Having done this, both parties can be sure that the other node holds the same secret, hence mutual authentication is accomplished.

Pairwise session key establishment is the second goal that our authentication protocol must achieve. That is at the end of a protocol run, the two participating parties must possess the same secret key, which is freshly generated. Messages three and four of the protocol are responsible for the key establishment. In the third packet, one node sends his Diffie-Hellman public components to the other node. After receiving the packet, the receiver will be able to compute a new key using the received information and his private components. This node then sends his Diffie-Hellman public components back to the sender of the third message. Upon the receipt of the message, a new key can be computed. By the definition of Diffie-Hellman key agreement, the two established keys will be the same. The simulation results confirm that the keys computed by two authenticating mobile nodes are indeed the same. Therefore, by using Diffie-Hellman keying mechanism as part of the authentication and key establishment protocol, we have achieved the second objective, which is the establishment of a new pairwise session key.

The final design criteria that the authentication protocol will be evaluated against is the efficiency, in terms of the performance. In Section 4.1.1, we state that we would like the protocol to involve as few messages as possible and to run as fast as possible, while also achieving the other two goals. In our authentication and key establishment protocol, four messages are exchanged between two mobile nodes. Within these four messages, we achieve mutual authentication and the establishment of a new pairwise session key as explained above. Excluding the initialisation stage, no matter which setting or which situation a mobile node is in, on average the time taken to complete the authentication and key establishment between two mobile nodes is approximately ten milliseconds. The time taken only increases linearly as the number of mobile nodes authenticating with one mobile node increases.

However, when a private local mobile ad hoc network is first formed, our results suggest that the time taken for all the nodes (if they are all within each other's radius) to authenticate and establish a new pairwise key with one another increases quadratically as the number of mobile nodes increases. This is admittedly the negative point of our protocol in that there will be a noticeable delay if and when there are more and more mobile nodes within the coverage of one another. Nevertheless, the network setup only has to be done once, and all the mobile nodes will be able to communicate with one another securely due to the keys that have been established as a result of the protocol run(s).

Once the network is set up, the time taken for a mobile node to complete the authentication when joining the network will increase linearly as the number of mobile nodes that it can reach increases. Even though this is the case, the results appear to be satisfactory in terms of the speed of the protocol, especially in a one-to-one situation which takes around ten milliseconds only. Even if a joining node can reach around 140 mobile nodes, it would take less than one second to complete the authentication as well as establish a new shared secret with all of them in the outdoor environment, according to our Equation 6.1. This situation (having more than 140 reachable nodes) in the indoor environment is very unlikely, considering the maximum range of each of the mobile nodes, as we have found in the network simulator, is only about five to seven metres. On the whole, the time taken for a mobile node or mobile nodes to join an existing ad hoc network, whether it is a one-to-one, one-to-many or many-to-one situation, is very little when we think about the work that the protocol accomplishes (i.e. mutual authentication and the establishment of a new key with every reachable nodes) after the protocol has finished.

It appears that the authentication and key establishment protocol has met all of the criteria set in Section 4.1.1. Now we will analyse it to see whether or not our protocol can overcome the problems of the existing protocols (see Section 3.1).

The advantages that our authentication and key establishment protocol has over the existing protocols are as follows. First of all, the main problem with ID-based cryptography is that a mobile node has to contact a private key generator in order to get a decryption key. It is also not possible for each node to compute his own key, and achieve a secure network. Threshold cryptography has a similar problem in that the mobile node has to contact a number of machines in order to obtain the private key for decrypting the message. Our protocol overcomes this problem by integrating the Diffie-Hellman key agreement as part of the authentication mechanism. As a result, each party will hold the keys necessary for decrypting the messages received, hence there is no need to contact any third party in order to obtain the key(s). Moreover, having each mobile node hold a pairwise key for each of his neighbours does not increase the vulnerability or decrease the security of the network. It actually improves the privacy of the communication between two parties in that only those parties that hold the correct key can decrypt the message. By not having to contact any other parties, our protocol provides better efficiency, flexibility and scalability as a result.

Secondly, our protocol has solved the problem of a single point of failure, which arises in cluster based authentication. Our authentication protocol has each node working independently of one another. Each of the mobile nodes carries out the authentication and keying mechanism without relying on any external third party. This means that even if a mobile node is compromised the mobile ad hoc network will still function.

Thirdly, the problem of zero-knowledge based authentication explained in [WYOP94] is that there is a lack of mutual authentication. Our authentication protocol mitigates this in the first three messages of the protocol, as explained above.

Fourthly, the problem with the resurrecting duckling techniques and location-limited authentication is that the two authenticating parties must be very close to one another to carry out the protocol securely. When using our protocol, however, mobile nodes do not even have to be in the line-of-sight of one another to carry out the protocol, as long as they are within each other's radius. Our protocol is also secure according to the analyses and proofs done in Sections 4.1.2 and 4.1.3. This implies that our authentication protocol provides better flexibility and scalability. The ability to operate the protocol without "seeing" other mobile nodes is also an advantage over the secure spontaneous interaction method, which needs some human interaction when executing the protocol (e.g. looking at flashing lights or listening to a musical tune). This brings us to another advantage of our protocol which is there is no need for any human interaction when carrying out our protocol, as everything is processed in accordance with the protocol description.

On the whole, our authentication and key establishment protocol has addressed all the problems created or not solved by any of the existing protocols. This includes the ability for each mobile node to compute his own decryption key(s), the ability for each node to work without relying on one another, the ability for the protocol to work in a large scale and the ability of the protocol to work without any human interaction. Moreover, our authentication and key establishment protocol appears to fall under the authentication class known as the homogeneous class of authentication protocols [ARE$^+$05]. The reason for this is that all mobile nodes in the network have the same role with respect to the authentication operation.

In general, the authentication and key establishment protocol has met all of the design criteria that are set in Section 4.1.1. The protocol can achieve mutual authentication, pairwise key establishment between two mobile nodes, and efficiency to a certain degree. Our protocol has also overcome the problems that are present in the existing protocols. However, there is one drawback to the protocol. That is, the time it takes to set up a network depends on how many mobile nodes are within each other's coverage. The more mobile nodes there are, the longer it would take, as seen in the quadratic relationship explained above. One might also say that because a shared secret, K, must be known and used by all the parties when joining the network, there is a risk that if the secret is compromised then the whole network could also be compromised. We could argue that because this protocol is designed to work within a private local mobile ad hoc network, the shared secret is a must-have, so that any node that would like to join could be verified that he or she is authorised to be part of the network.

We believe that the authentication and key establishment protocol provides sufficient security as seen in the proofs and analyses in Chapter 4. Moreover, as the results of the simulations suggest, the protocol is efficient enough in terms of the performance to be employed in private local mobile ad hoc networks.

The next section provides the results and evaluations of the pre-handoff, i.e. what happens after a network is formed or after a new node has joined the network.

6.3 Pre-handoff

The results of the simulations are presented in Table 6.6. The table shows how long it takes one mobile node to process the security information of n mobile nodes, after Hello messages have been exchanged and the multipoint relays have been selected. By "processing" information, we mean receiving and storing the RSA public components of each node, and computing new Diffie-Hellman components, so that the node is ready for the handoff protocol. The results are then plotted and shown in Figure 6-5.

Note that each of the times given in the table is the average time obtained from running each simulation ten times. Also note that we only concentrate on the performance of this stage of the protocol, because the exchange of Hello messages and the topology information updates are already parts of most routing protocols, and the performance of the multipoint relay algorithm can be found in [QVL00].

6.3.1 Evaluation

The aim of the pre-handoff stage is to get mobile nodes ready for the handoff process. This stage involves the exchange of Hello messages, the selection of multipoint relays and the security information processing.

As explained in Chapter 4, the multipoint relay algorithm is applied so that a

Table 6.6: Time taken to process security information

Number of nodes	Time (s)
1	0.0000
5	0.0031
10	0.0113
20	0.0141
30	0.0227
40	0.0305
50	0.0379
60	0.0492
70	0.0504
80	0.0515
90	0.0535
100	0.0613
110	0.0660
120	0.0730
130	0.0758
140	0.0824
150	0.0973

mobile node can select a set of one-hop neighbours to pass his security information to his two-hop neighbours. In using the multipoint relay algorithm, the time taken to relay messages to the two-hop neighbours of any mobile node can be reduced approximately by half when compared to the normal flooding technique (according to the findings in [QVL00]). Moreover, multipoint relaying creates much less traffic in the mobile ad hoc network, which has bandwidth limitations, as mentioned in the earlier chapters. By generating less traffic and taking less time to send messages, we claim that the multipoint relay algorithm can help overcome some of the mobile ad hoc network limitations - including bandwidth limitations, computational power limitations and battery life limitations.

The simulations test how long it would take one mobile node to process the security information of his two-hop neighbours received from the selected multipoint relays. The simulations are run with the number of two-hop neighbours ranging from one to one-hundred and fifty nodes. The results and graph suggest that there is approximately a linear relationship between the number of nodes and the time taken to process the security information. Our findings show that

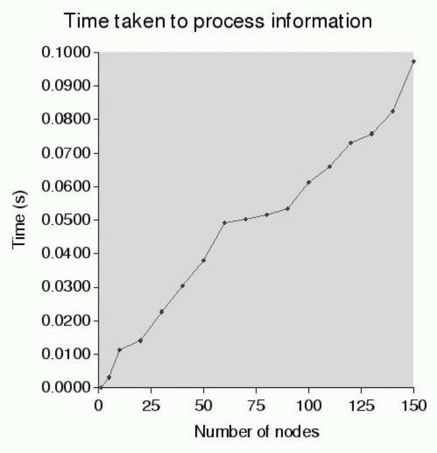


Figure 6-5: Time taken to process security information

it takes less than 0.1 seconds to process the information of one-hundred and fifty mobile nodes. This is significant because this step has to be completed before a handoff process can begin, it is important that the information is processed quickly. In processing the information for one-hundred and fifty nodes in less than 0.1 seconds, we have demonstrated that it is possible to complete this step before a handoff protocol is initiated (although how much information can be processed also depends on the speed of the roaming node as will be seen in the next section).

The selection of multipoint relays and the processing of security information of mobile nodes constitute a large part of the pre-handoff stage. We have not mentioned the performance of the exchange of Hello messages and the updates of topology tables and neighbour lists, because most routing protocols use these mechanisms anyway. The multipoint relay algorithm is useful in this stage because it helps to reduce the amount of traffic within the network. We have also demonstrated that it is possible to complete the security information processing within a very short period of time.

6.4 Bringing the two together

We have already illustrated that separately the authentication and key establishment and the pre-handoff protocol function as we expect. This time, we combine the two protocols in larger networks, which consist of twenty mobile nodes, fifty mobile nodes, one hundred nodes, one hundred and fifty nodes, two hundred nodes and two hundred and fifty mobile nodes. All of the network layouts can be found in Appendix B.

One significant outcome that we have found during the simulations is that as the network becomes denser (i.e. there are more mobile nodes within the network), there seems to be a more noticeable delay before the authentication protocol and the pre-handoff protocol are completed.

On the other hand, it appears that the simulations give the results as expected. That is, the correct topological information is produced after the authentication has been accomplished as well as after the pre-handoff protocol. As a result of the pre-handoff protocol, the mobile nodes process and store the security information correctly. Also when "adding" new mobile nodes to the network, the authentication and key establishment for each mobile node behaves in one of the four scenarios that we have tested earlier. The pre-handoff stage also behaves as our protocol description states when there is a change in the network topology.

6.4.1 Evaluation

As we have mentioned before, the main purpose of the simulations in this section is that we would like to demonstrate that our authentication protocol and the pre-handoff stage are able to cope with a larger number of mobile nodes within larger settings.

For the authentication protocol, specifically when the network is first set up, in these larger settings, the time taken to complete is approximately equal to that of a "sub-network" with the biggest number of mobile nodes that can reach one another. The reason that it takes longer when there are more mobile nodes lies within the scenario 4 of the authentication simulations (or many-to-many authentications) as well as the fact that the denser the network, the more mobile nodes will be within each other's radius. In other words, as Equation 6.3 suggests, the more mobile nodes there are that are within each other's coverage, the longer it will take to complete the authentication and key establishment.

Similarly, for the pre-handoff stage in the larger settings, the time taken to complete this stage is approximately equal to the time it takes a mobile node, which has the most security information to process (i.e. the node that has the biggest number of two-hop neighbours) to complete the pre-handoff stage. Again, if the density of the network is greater, each mobile node should have more neighbours, which means that his one-hop neighbour should also have more neighbours. Those neighbours would then become other nodes' two-hop neighbours. Therefore, as it is shown in the previous section, the bigger the number of two-hop neighbours, the longer it will take to complete the pre-handoff stage.

From the simulations in this section, it is shown that our authentication and pre-handoff protocols are able to function in larger and more complex settings as well as smaller and less complex ones (as shown in previous sections).

Once mobile nodes have completed the authentication and pre-handoff protocols, they are ready to initiate or handle the handoff process, whose results of the simulations will be shown in the next section.

6.5 Handoff

We divide the results into several sections according to the settings of the simulations - outdoor environment (single handoffs), indoor environment (single handoffs), simultaneous (multiple) handoffs and handoffs with non-stationary mobile nodes.

6.5.1 Outdoor environment (single handoffs)

The results presented here are the average times that are obtained from running each particular setting ten times. The speed of the roaming node is varied from the average walking speed to the speed of a Eurostar train. The results are presented in Table 6.7.

In the Table 6.7, 1.34 ms^{-1} is the average walking speed, 5.00 ms^{-1} is the average jogging/running speed, 13.4 ms^{-1} is the speed of a node traveling at 30 mph, 26.8 ms^{-1} is the speed of a mobile node travelling at 60 mph and 83 ms^{-1}

(or 186 mph) is the average speed of the Eurostar.

Note that each of the times in the Table 6.7 is the total amount of time starting from when the disassociation_wait (or pre-disassociation) packet is sent until the disassociation message is received by the old mobile node (i.e. the node that the roaming node has moved away from).

Table 6.7: Time taken to complete the handoff protocol

Speed (ms^{-1})	Time (s)
1.34	0.0102
5.00	0.0093
13.4	0.0097
26.8	0.0092
83.0	0.0094

From the results that we have acquired, the average time taken to complete the handoff process is 0.00956 seconds or 9.56 milliseconds.

The second simulation, explained in Section 5.4.1, is to determine the maximum number of two-hop neighbours the roaming node can have, so that he can complete the pre-handoff phase before initiating a new handoff with any of those two-hop neighbours (in case the handoffs are to happen in succession). The Table 6.8 shows the results of the simulations.

Table 6.8: The maximum number of two-hop neighbours, whose information the roaming node can process prior to a next handoff

Speed (ms^{-1})	Number of nodes
1.34	560
5.00	170
13.4	67
26.8	33
83.0	11

6.5.2 Indoor environment (single handoffs)

This section presents the results of the handoff protocol simulated in an indoor/office environment. These simulations include the special scenarios (explained in Section 5.4.2) as well. Table 6.9 shows the time taken to complete a handoff process in the following scenarios. Simulation 1 is where a mobile node moves in a straight line (which could be thought of as moving along a straight corridor), simulations 2, 3, 4 and 5 are where a mobile node moves diagonally (which could be thought of as a mobile node moving past a corner in a building) in different network setups. Each of the simulations was run ten times. The average time for each of them is recorded in the Table 6.9.

Table 6.9: Time taken to complete the handoff process indoor

Scenario	Time (s)
1	0.0115
2	0.0287
3	0.0134
4	0.0098
5	0.0261

From the scenarios 1 to 5, the results we have obtained suggest that on average the time taken to complete the handoff process in an indoor environment is approximately 0.0179 seconds or 17.90 milliseconds.

Special cases

For the special cases, there are two methods that allow us to accomplish a handoff. They include the handoff protocol with additional messages and the re-authentication protocol. Both methods are explained in Section 5.4.2. Each of the simulations was run ten times. The average times of the simulations are presented in the Table 6.10. Figure 6-6 shows a bar chart comparing the time taken to complete a handoff process using the two different approaches.

The results that are shown in Table 6.10 suggest that if the handoff protocol with extra messages is used in the special cases, the average time it takes to complete a handoff is approximately 0.0267 seconds or 26.7 milliseconds. If the re-authentication process is carried out, however, the average time is approximately 0.0215 seconds or 21.5 milliseconds.

Table 6.10: Time taken to complete the special case of handoff

Method	Time (s)
Handoff (1)	0.0338
Handoff (2)	0.0232
Handoff (3)	0.0231
Re-authentication (1)	0.0318
Re-authentication (2)	0.0164
Re-authentication (3)	0.0173

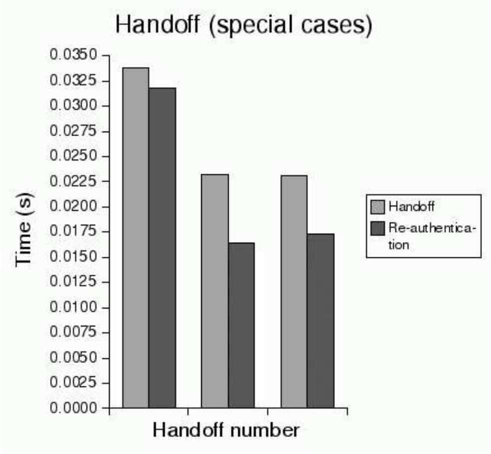


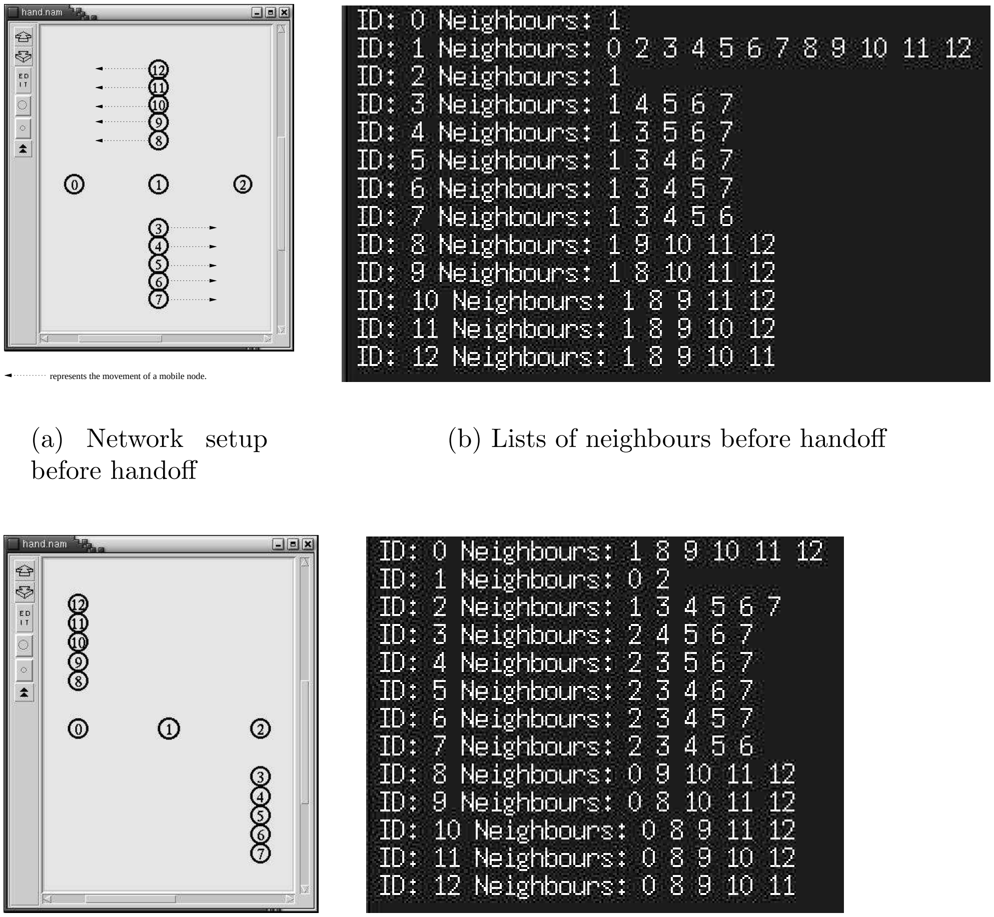
Figure 6-6: Time taken to complete a handoff in special cases

6.5.3 Simultaneous handoffs

The main purpose of these simulations is that we would like to demonstrate whether or not our handoff protocol will be able to handle the situation where multiple handoffs happen simultaneously.

The results of the simulations show that the mobile nodes, which roam to either Node_(0) or Node_(2), update their neighbour lists and topology tables, and are added to the neighbour list and the topology table of the "destination" node correctly. They are also correctly disassociated from Node_(1), and vice versa.

Figure 6-7 shows simultaneous handoffs that take place in a sample network. Figures 6-7(a) and 6-7(b) display the setup of the sample network together with an extract from our protocol output showing the list of one-hop neighbours of each mobile node, after the authentication protocol has been carried out. Figure 6-7(c) provides the setup of the network after the mobile nodes have moved. Figure 6-7(d) provides the list of one-hop neighbours of each mobile node, after the handoffs have been carried out simultaneously by the roaming mobile nodes. From the extract of the protocol output (as well as the results of other simulations), it can be seen that the mobile nodes and the handoff protocol are able to handle simultaneous handoffs.



```
ID: 0 Neighbours: 1
ID: 1 Neighbours: 0 2 3 4 5 6 7 8 9 10 11 12
ID: 2 Neighbours: 1
ID: 3 Neighbours: 1 4 5 6 7
ID: 4 Neighbours: 1 3 5 6 7
ID: 5 Neighbours: 1 3 4 6 7
ID: 6 Neighbours: 1 3 4 5 7
ID: 7 Neighbours: 1 3 4 5 6
ID: 8 Neighbours: 1 9 10 11 12
ID: 9 Neighbours: 1 8 10 11 12
ID: 10 Neighbours: 1 8 9 11 12
ID: 11 Neighbours: 1 8 9 10 12
ID: 12 Neighbours: 1 8 9 10 11
```

(a) Network setup before handoff

(b) Lists of neighbours before handoff

```
ID: 0 Neighbours: 1 8 9 10 11 12
ID: 1 Neighbours: 0 2
ID: 2 Neighbours: 1 3 4 5 6 7
ID: 3 Neighbours: 2 4 5 6 7
ID: 4 Neighbours: 2 3 5 6 7
ID: 5 Neighbours: 2 3 4 6 7
ID: 6 Neighbours: 2 3 4 5 7
ID: 7 Neighbours: 2 3 4 5 6
ID: 8 Neighbours: 0 9 10 11 12
ID: 9 Neighbours: 0 8 10 11 12
ID: 10 Neighbours: 0 8 9 11 12
ID: 11 Neighbours: 0 8 9 10 12
ID: 12 Neighbours: 0 8 9 10 11
```

(c) Network setup after handoff

(d) Lists of neighbours after handoff

Figure 6-7: Results of simultaneous handoffs

6.5.4 Handoffs with non-stationary nodes

As we have explained before, the main aim of running these simulations in this section is to illustrate what happens when a mobile node roams away from another mobile node, which is also moving, but in a different direction. That is we would like to see whether the nodes would be able to operate the handoff protocol and the disassociation process properly.

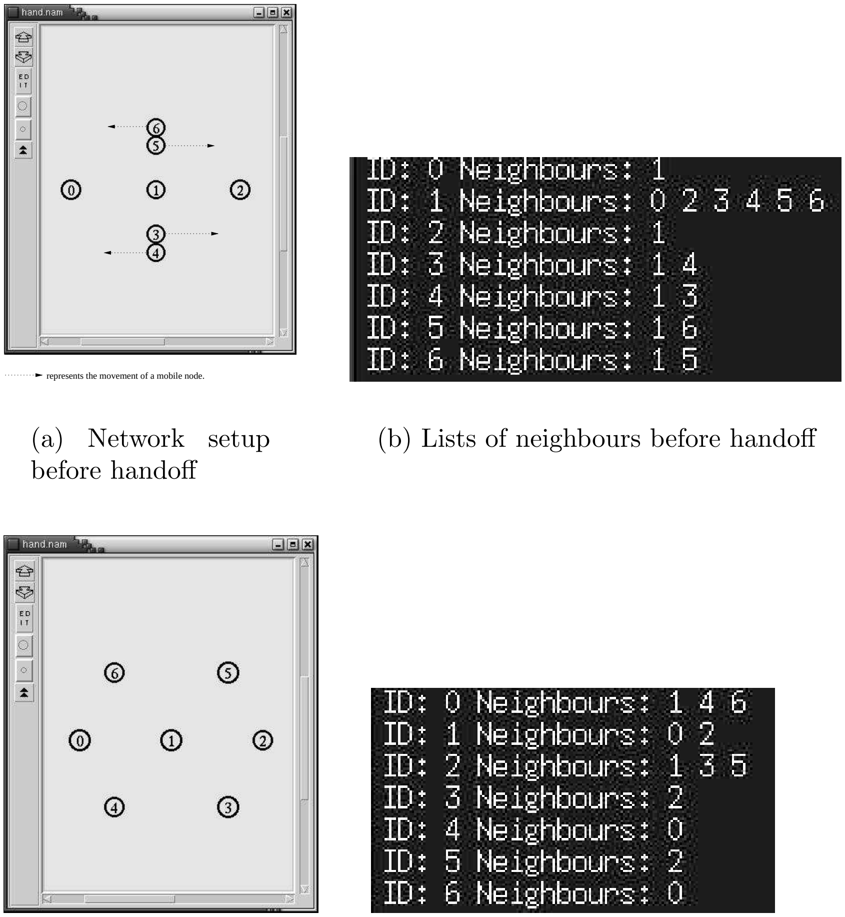
We have obtained the results, which suggest that mobile nodes carry out the handoff protocol correctly as well as properly disassociate themselves with both the other moving mobile nodes and the ones that are stationary.

Figure 6-8 shows the handoffs that take place in a sample network. Figures 6-8(a) and 6-8(b) display the setup of the sample network together with an extract from our protocol output showing the list of one-hop neighbours of each mobile node, after the authentication protocol has been carried out. Figure 6-8(c) provides the diagram of the network after the four mobile nodes have roamed. Figure 6-8(d) provides the list of one-hop neighbours of each mobile node, after the handoff protocol. From the extract of the protocol output (as well as the results of other simulations), we can see that the mobile nodes managed to carry out the handoff protocol as well as the disassociation process correctly.

6.5.5 Evaluation

We have presented the results of the simulations for the handoff protocol in both outdoor environment (where the line-of-sight between mobile nodes is clear) and indoor environment (where nodes are separated by hard partitions, hence the line-of-sight between nodes is obstructed and the coverage of a node is not always a perfect circle), single and multiple handoffs, and handoffs with stationary and non-stationary mobile nodes. In this section, the results will be analysed, and we will explain whether our handoff protocol satisfies the requirements set in Section 4.3.1 and whether it solves the problems, explained in Section 3.2, that the existing handoff protocols have.

Once the pre-handoff stage is finished, mobile nodes will be ready to handle the handoff process. For the outdoor environment, our results show that the average time taken to complete the handoff protocol is approximately 0.00956 seconds or 9.56 milliseconds.



```
ID: 0 Neighbours: 1
ID: 1 Neighbours: 0 2 3 4 5 6
ID: 2 Neighbours: 1
ID: 3 Neighbours: 1 4
ID: 4 Neighbours: 1 3
ID: 5 Neighbours: 1 6
ID: 6 Neighbours: 1 5
```

(a) Network setup before handoff

(b) Lists of neighbours before handoff

```
ID: 0 Neighbours: 1 4 6
ID: 1 Neighbours: 0 2
ID: 2 Neighbours: 1 3 5
ID: 3 Neighbours: 2
ID: 4 Neighbours: 0
ID: 5 Neighbours: 2
ID: 6 Neighbours: 0
```

(c) Network setup after handoff

(d) Lists of neighbours after handoff

Figure 6-8: Results of a sample simulation

The results of the simulations that are set in the outdoor environment indicate that no matter how fast the roaming node travels, it takes roughly the same amount of time to complete the handoff protocol. We have shown by running these simulations that our handoff protocol is able to cope with variety of speeds of the roaming node, ranging from the walking speed up to the speed of Eurostar.

The results of the next simulations suggest that there is a limit of how many two-hop neighbours a travelling node can have while roaming. In details (with the help of Figure 5-5), once Node_(3) completes the handoff process, by the mechanism of any routing protocol the Hello messages will be exchanged (due to the change in network topology) in order to update the topology information. In addition, our pre-handoff phase states that the roaming node should have all of the security information of the two-hop neighbours ready for the future handoffs. With reference to Figure 5-5, this means that Node_(3) will have had to process the information of all of the Node_(1)'s one-hop neighbours before he can carry out the next handoff.

In the case where the node never stops moving, our results show that if a mobile node is travelling at 1.34 ms^{-1} or at an average walking speed, the node can handle approximately up to 560 two-hop neighbours. That means the roaming node can process the information of 560 mobile nodes before starting the next handoff. When a roaming node travels at 5 ms^{-1} or at a running speed, the maximum number of nodes that he can process before the next handoff is approximately 170 nodes. When travelling at 30 mph and 60 mph, a node can handle 67 nodes and 33 nodes respectively. If a node travels at 83 ms^{-1} or at the average speed of the Eurostar, he can process the information of 11 other nodes before initiating the next handoff. These results indicate that the faster a node travels, the less information he can process in time for the next handoff protocol. The main purpose of running these simulations is that we would like to know the limitation of the handoff protocol if it were to happen in succession.

For the handoff protocol in the indoor settings, we will analyse the results from the scenarios 1 to 5 (handoffs with no extra messages) separately from the special cases. The time taken to complete the handoff protocol in the scenarios 1 to 5 tend to vary from approximately 10 milliseconds to 29 milliseconds, which makes the average time approximately 17.90 milliseconds. The results are as expected due to the properties of the indoor environment. That is each of the mobile nodes is separated from one another by hard partitions, and the coverage of each mobile node is not always a perfect circle. This means that the strength of the signal, the ability to receive the signal and the sensitivity of the antenna of any one mobile node is not constant around the node. Furthermore, because of the partitions the signal strength is sometimes unpredictable, i.e. the signal may be stronger in one place but considerably weaker in another. Another factor that contributes to the variation of the time taken to complete the handoff process is the amount of time it takes a disassociation message to travel from the roaming mobile node to the node that he/she is moving away from.

For the special cases, where the signal from a three-hop neighbour is stronger than that from a two-hop neighbour. The problem with this is that by the mechanism of our pre-handoff protocol, a roaming node only has the information to carry out the handoff protocol with his two-hop neighbours. However, it has been demonstrated by the simulations that these special cases can be handled by both our handoff protocol with additional messages and the re-authentication method.

The simulation results suggest that the average time taken to complete the handoff protocol in the special cases using our handoff protocol with additional messages is approximately 26.70 milliseconds. The time taken is longer than that of the normal cases. This is as we expected because there are four extra messages exchanged before the protocol can be completed. With reference to Figure 5-7 as an example, those four messages are that Node_(3) has to obtain Node_(0)'s information from Node_(2), and Node_(0) has to obtain Node_(3)'s information from Node_(1). This can be accomplished because Node_(0) is Node_(2)'s two-hop neighbour and Node_(3) is Node_(1)'s two-hop neighbour. By completing the pre-handoff protocol prior to the handoff, they should have the necessary information to pass on to Node_(3) and Node_(0).

The other method that has been simulated for the special cases is the re-authentication. The simulation results show that on average, it takes approximately 21.50 milliseconds to complete a handoff. The time taken to complete a special case of a handoff is not as long as that of the normal handoff protocol due to the smaller number of messages being exchanged during the process. That is, seven messages are needed for this re-authentication method rather than nine messages for the handoff protocol.

At this stage, it seems appropriate to remind ourselves that the first message of the authentication protocol, which is also the first message of the re-authentication process, is encrypted using the key, K, shared by all members of a private mobile ad hoc network. For the special cases, even though it takes less time for the re-authentication method to complete a handoff process, we would still recommend that the usual handoff protocol with additional messages should be employed. This is because we would like to avoid the constant re-use of the shared secret, K. The use of the shared secret, K, should be restricted to authentication purpose when a mobile node first joins the network only. If the re-authentication method were used during a handoff process, the constant re-use of the shared key could make the wireless network more vulnerable in that an adversary could carry out statistical attacks to recover plaintext.

In addition, we have also tested two further scenarios, including simultaneous handoffs and the situation where mobile nodes move away from each other into the coverage of different nodes. We have shown that our handoff protocol is able to cope with more than one handoff at any one time. We have also demonstrated that when mobile nodes move away from one another and enter the coverage of different mobile nodes our handoff protocol functions properly, and the nodes disassociate themselves from each other correctly, too (as shown by the list of neighbours of each mobile node in the protocol output).

From the handoff simulations that were run and the results that were obtained, we have shown that our handoff protocol is able to cope with both single and multiple handoffs as well as handoffs with both moving and non-moving mobile nodes.

We will now discuss whether the performance of the handoff protocol satisfies the criteria set in Section 4.3.1, and whether the protocol can overcome the problems, mentioned in Section 3.2, that the existing handoff protocols have.

The four criteria, stated in Section 5-7, that the handoff protocol must achieve are make-before-break, protocol efficiency , pairwise key establishment with the new node and mutual authentication.

First of all, the simulations have shown that our handoff protocol can achieve make-before-break in all of the scenarios that we have tested, even in the special case where extra messages are exchanged. That is the roaming node makes a new connection with the node he is approaching before losing the connection with the one he is moving away from. However, there is a limit to how many (new) two-hop neighbours a travelling node can have after the first handoff (in the case where the node keeps on travelling, more handoff processes will be necessary). By the description of our protocol(s), the roaming mobile node must have all of the necessary information of all of his two-hop neighbours before a handoff process can begin. This is important, because if all of the information is available, it will not matter which direction the node is moving and which party the node carries out the handoff protocol with. Even though there is a limit, the number of two-hop neighbours that a roaming node can have is still large (although the number decreases as the speed of the node increases). Despite the limitation, we claim that our handoff protocol can run quickly enough to satisfy the first design criteria, make-before-break.

Secondly, the travelling node must be able to establish a pairwise session key with the node that he is approaching. The key establishment is achieved during the exchange of the third and fourth messages of the protocol, namely the Handoff_Request and Handoff_Return. The Diffie-Hellman key agreement method is used to accomplish this task. By the definition of Diffie-Hellman key agreement, the two parties should end up with the same key. The simulation results have shown that the keys established by the roaming node and the "new" node are indeed the same. Therefore, by incorporating this keying mechanism into our handoff protocol, we have achieved our objective in having two mobile nodes compute a new pairwise session key.

Protocol efficiency is the third goal that our handoff protocol must achieve. The results of the simulations in both outdoor and indoor settings show that it takes approximately ten and twenty milliseconds respectively to complete the handoff protocol. This is less than the time recommended by the International Telecommunication Union or ITU. The ITU guidance states that the total handoff latency should be less than fifty milliseconds. This claim is also supported by Ben Guderian, the director of marketing at SpectraLink, who says "As you go above 50 milliseconds, you start to run into situations where you have to have larger buffers in order to compensate for the fact that there's that variability in how long the handoff is going to take. What happens when you have larger buffers is that adds delay into the system. When you get above 50 to around 100 milliseconds, it's a noticeable delay, kind of like what we used to deal with satellite telephone calls." On the whole, five messages (nine messages in the special cases) are exchanged between nodes during the handoff process. Within that time, the roaming node successfully completes the handoff, disassociates himself with the "old" node and establishes a new pairwise key with the "new" node. On average, this can be done more than twice as fast as the time recommended by the ITU. One factor that contributes to the fast performance of the handoff protocol is that the calculations of Diffie-Hellman components (needed for key establishment) have been done during the pre-handoff protocol. We, therefore, claim that, in terms of the performance, our handoff protocol suffices the efficiency criteria.

The final criteria to be met is mutual authentication between the roaming node and the node he/she is approaching. This is achieved during the exchange of the actual handoff messages, i.e. messages 3 and 4 of the protocol. That is, the travelling node sends a *Handoff_Request* message to the node he/she is approaching together with his/her signature. Upon receiving the message, the new node checks the identity of the sender in the existing topology table as well as verifying the sender's signature. If the sender exists as a member of the network and the signature verification succeeds, the new node sends a reply message saying that the handoff process is allowed. When the reply packet reaches the roaming node, he/she does the same checking and signature verification process. By doing this, mutual verification is accomplished.

In addition to those four criteria that the protocol has so far satisfied, there is another feature that our handoff protocol has achieved. That is *pre-authentication*. What we mean by pre-authentication of handoff is that when roaming, the travelling node does not have to be fully authenticated again. The authenticity and identity of the roaming node and the node that is being approached are verified by using the public key cryptosystem (i.e. digital signature) as well as the existence of the node in the topological information. This is possible due to the exchange of messages and information processing that has been done during the pre-handoff stage prior to the current handoff. This pre-authentication feature is another reason that contributes to the fact that our handoff protocol takes less time than what the ITU recommends.

It appears that the handoff protocol has met all of the criteria set in Section 4.3.1. Now we will analyse the protocol to see whether or not it can overcome the problems of the existing protocols (see Section 3.2).

The main problem that the existing handoff protocols have is that they are more suited to solving the macro-mobility problem (movement of mobile nodes between domains) or handoff in infrastructure wireless networks, whereas our work focuses on the micro-mobility problem (movement of mobile nodes within one domain) and handoff in pure mobile ad hoc networks. The existing protocols including [Per96, PC02a, PC02b, TLP99, CP96, MSA03, ABABD03], as explained in Section 3.2, are not specifically designed to work within mobile ad hoc networks. They often involve other third party hardware, such as home and foreign agents, and/or access points. Some of the protocols even go as far as having access points decide when mobile nodes should begin a handoff process. Our goal, therefore, is to solve these problems by eliminating the involvement of any third party and having the roaming mobile nodes make their own decision of when to initiate the handoff protocol.

Firstly, as the description of our handoff protocol says, the handoff process will be initiated if the roaming node receives a stronger signal from some node other than the one(s) he is currently associated to. From this account, it can be seen that the roaming mobile node is the one who makes the handoff decision rather than relying on other party to make it for him. Secondly, the handoff protocol is designed specifically for pure mobile ad hoc networks, which results in involving the mobile nodes that are parts of any particular handoff only. In general, only the roaming mobile node and the node that is being approached are involved in the actual handoff process. The node that the travelling node is moving away from is part of the protocol for the disassociation purpose only.

One may argue that our handoff protocol is somewhat similar to the handoff protocols that are designed for infrastructure wireless networks, such as [MSA03] and [ABABD03], in that information is transferred among the old and new access points, and the travelling mobile node itself. However, the performance of our protocol has shown that the time taken to complete our handoff protocol is significantly lower. In the case of [MSA03], on average their protocol takes approximately ten to twenty times longer than our protocol, about two hundred to four hundred milliseconds. This fact indicates that our protocol has indeed improved the efficiency of the existing handoff protocols, which further confirms that the protocol has achieved the efficiency required by the design criteria stated in Section 4.3.1.

On the whole, our handoff protocol has addressed the problems that the existing protocols have. We have overcome the problems by eliminating the extra third party hardware and having roaming mobile nodes make a handoff decision themselves. By doing that, the scheme has become more suited to pure mobile ad hoc networks.

The handoff protocol has met all of the design criteria stated in Section 4.3.1.

The protocol can achieve mutual authentication, make-before-break, pairwise key establishment with the approached node and protocol efficiency. Our protocol, as explained above, has also overcome the problems that the existing protocols have. Furthermore, we have shown that the performance of our handoff protocol is significantly better than that of the existing protocols (even though the existing handoff protocols are not designed for mobile ad hoc networks).

We believe that the handoff protocol provides sufficient security, as seen in the proofs and analyses given in Section 4.3. Moreover, as the results of the simulations suggest, the protocol is efficient in terms of performance, which makes it more suitable to be employed in a private local mobile ad hoc network.

From the descriptions and the performances of the authentication, pre-handoff and handoff protocols, it can be seen that all of them are closely related. First, when forming a mobile ad hoc network or joining an existing network, a mobile node broadcasts his RSA public components which are integrated into the request-to-join packet. All of the mobile nodes that "hear" the request packet will then hold this information. Next, during the pre-handoff stage, this security information is exchanged among mobile nodes (a mobile node, his selected multipoint relays and his two-hop neighbours). The same information (the RSA public components) is used for encryption, decryption and identification purposes during the handoff protocol. By doing this, as suggested before, there is no need for any node to be completely re-authenticated while carrying out the handoff process, i.e. pre-authentication.

Summary

This chapter has provided the results and the evaluations of the performances of the authentication and key establishment protocol, the pre-handoff stage and the handoff protocol. We have demonstrated that the authentication protocol has met all of the design criteria which include mutual authentication, pairwise session key establishment and efficiency. We have also shown how our protocol is able to mitigate the problems that the existing protocols have. The results suggest that on average a one-to-one authentication and key establishment takes approximately ten milliseconds to complete. The time increases linearly in the one-to-many and many-to-one situations. Furthermore, we have found that in a many-to-many scenario (or when a network is being set up for the first time), the time taken to complete this protocol increases quadratically as the number of mobile nodes increases. Note that this is the case only if and when all of the mobile nodes are within each other's coverage.

The pre-handoff stage applies the multipoint relay algorithm which helps reduce the traffic and the time it takes for all of the two-hop neighbours of a node and that particular node to receive the security information. Hence it assists in overcoming the limitations of mobile ad hoc networks, including the bandwidth limitations. The time taken to process the security information increases linearly as the number of nodes increases. Even that is the case, the results show that this stage can be completed quickly.

For the handoff protocol , our results show that it can be completed in between ten and twenty-six milliseconds, depending on the environment the network is in at the time. There is, however, a limit to how many new two-hop neighbours the roaming node can have after the first handoff. This is because the information of all the two-hop neighbours has to be processed before the next handoff can begin. Even though there is a limit, our results show that it should still be adequate for any mobile ad hoc networks. For the special cases of handoff, we have explained that despite the shorter time that the re-authentication process takes, we would choose the handoff protocol plus the additional messages over it. The reason is that the constant re-use of the shared secret, K, should be avoided. We have also run the simulations to show that as well as single handoffs, simultaneous handoffs can be correctly handled. In addition, mobile nodes, when applying our handoff protocol, appear to have the ability to cope with handoffs with both stationary and non-stationary mobile nodes. Furthermore, we have shown that the handoff protocol has met all of the design requirements stated in Section 4.3.1. We have also explained how our protocol would perform better in mobile ad hoc networks than the existing handoff protocols.

Moreover, we can see from the descriptions and simulation results of the pre-handoff and handoff protocols that wherever a mobile node moves to, all of the nodes within the two-hop radius will always be able to handle the handoff process.

From the results and evaluations of the performances of the protocols, we can claim that we have designed and developed an authentication protocol, a pre-handoff protocol and a handoff protocol that are suited and applicable to private local (pure) mobile ad hoc networks.

Here we have shown and evaluated the performances of the authentication protocol, the pre-handoff stage and the handoff protocol. In the next chapter, we will conclude the dissertation.

Chapter 7

Conclusion and Future Work

In this last chapter, we will give a summary of the contributions of this work. Specifically, we look at the three protocols that we have designed and developed. They will be examined to see whether they have satisfied the security properties set in Chapter 1. We will again state the problems that we faced when developing these protocols for mobile ad hoc networks. At the end of the chapter, we outline several possible directions of future work.

7.1 Conclusion

A mobile ad hoc network (MANET) is a relatively new networking paradigm. It is different from the infrastructure wireless networks in that mobile nodes communicate *directly* with their neighbours via radio signals. Mobile ad hoc networks, being a type of wireless networks, are more exposed to security threats than the traditional wired networks. This is why the introduction of security protocols, such as authentication and secure handoff protocols, is necessary. This is especially the case for private local mobile ad hoc networks.

Our work is an attempt to produce a secure and efficient authentication protocol as well as a secure and efficient handoff protocol for private local mobile ad hoc networks. It aims at developing protocols which take the restrictions and limitations of pure mobile ad hoc networks - the lack of central authority, limited network bandwidth, computational power limitations, battery life limitations and physical security limitations - into consideration. The guiding principle of our protocols is that the protocols must be secure and efficient. That is, both the authentication and handoff protocols must satisfy security properties, including confidentiality, integrity, authenticity, availability and non-repudiation. Moreover, with regard to the physical restrictions of pure mobile ad hoc networks, the protocols must be able to accomplish their tasks quickly with the smallest number of packets possible and without involving any extra third party.

Designing and developing security protocols, specifically authentication and handoff protocols, for mobile ad hoc networks is more challenging, due to the physical limitations (e.g. the lack of fixed cables and the lack of central authority) and the lack of topological stability. The existing authentication and keying mechanisms that are currently employed in mobile ad hoc networks do not take these physical properties into consideration. Therefore, as we have explained in Chapter 3, they are not suitable to be employed in pure mobile ad hoc networks. For the handoff protocols, as far as we know, there are none that have been designed specifically for pure mobile ad hoc networks.

In this thesis, we have shown that there are performance and security problems with the existing authentication and handoff protocols which are currently employed in mobile ad hoc networks. Consequently, we have designed and developed an authentication and key establishment protocol, a pre-handoff protocol and a handoff protocol, all of which address the problems that the existing protocols have as well as take into account the physical constraints of mobile ad hoc networks.

We have also pointed out that the three protocols are closely related to one another. That is, the information used as part of the authentication protocol is passed among mobile nodes during the pre-handoff stage. The same information, namely the RSA public components, then becomes an integral part of the handoff protocol. Alternatively, the connection among the three protocols can be thought of as follows. After an authentication has been carried out, there is a change in network topology, which triggers mobile nodes to exchange Hello messages. We have extended this mechanism so that security information of mobile nodes as well as the topological information can be passed by the mobile nodes to one another. These processes constitute our pre-handoff protocol. As a result of running the pre-handoff, the mobile nodes should be ready to initiate or handle any handoff that may take place. After a handoff process has taken place, there is again a change in network topology, which again triggers the exchange of Hello messages and the pre-handoff protocol. The latter view of the relationship among our three protocols is depicted in Figure 7-1.

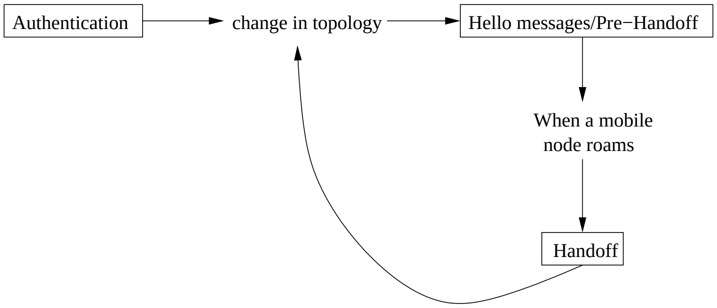


Figure 7-1: How the protocols are related

We have claimed that the authentication and handoff protocols are secure in unauthenticated-links model, according to the proofs and analyses using the BCK and CK methods. The correctness of the protocols has also been proved using the GNY logic. When carrying out the analyses and proofs, our thought was confirmed that the BCK and CK models must be used in conjunction with the GNY analysis. This is because the BCK and CK approaches alone only ensure that the communication channel between two parties is secure, and the GNY analysis helps reassure that any essential components that constitute a protocol message are not missing (which may make the protocol insecure).

In claiming that the authentication and handoff protocols are secure, they have achieved the security properties - confidentiality, integrity, authenticity, non-repudiation and availability - as follows. The protocols achieve confidentiality in that all of the messages are encrypted by either asymmetric encryption (RSA) or symmetric encryption (shared secret). Furthermore, as a result of each of the protocol run (authentication or handoff), the parties involved will have established a new pairwise key, which will be used as an encryption key in further communications. Both protocols achieve integrity in that message authentication code (MAC) or header checksum are integrated into all of the messages to ensure that it is possible to detect if and when the messages have been tampered with. Authenticity and non-repudiation are achieved by several means, including the use of public-key cryptosystem (digital signatures) and random nonces for proving the sender's identity. Lastly, availability is accomplished due to the following reasons. First, denial of service is (partly) prevented by having mobile nodes quietly discard all the packets (except for the request-to-join packet) sent by any nodes that have not yet been authenticated or part of the network. Secondly, since mobile nodes do not have rely on any other nodes to function, they will still be able to function even if others are compromised. Moreover, as explained in Chapter 6, both authentication and handoff protocols also achieve forward and backward secrecy as no information is used to establish a new shared secret more than once.

In addition to the security, due to the physical limitations of mobile ad hoc networks, the efficiency of the protocols in terms of performances is another important issue. We have demonstrated in our ns2 simulations, in Chapters 5 and 6, that the authentication and handoff protocols are able to perform efficiently. The results have shown that it only takes around ten milliseconds to complete a one-to-one authentication, and approximately only ten milliseconds to complete a handoff process in the outdoor environment, and twenty milliseconds in the indoor environment. We have achieved this by restricting the number of protocol messages to the smallest possible when carrying out the authentication and handoff. That is, only four messages for the authentication protocol and five messages for the handoff protocol. The authentication and handoff protocols are constructed by using a combination of well known cryptographic tools, namely RSA public-key cryptosystem and Diffie-Hellman key agreement. Moreover, by applying the multipoint relay algorithm to the pre-handoff stage, the traffic or the use of network bandwidth is reduced approximately by half.

However, there are a couple of limitations to our protocols. Firstly, there is one drawback to our authentication protocol when many-to-many authentications occur. That is, when every mobile node is within each other's radius, and they carry out the authentication and key establishment with one another simultaneously. The time taken to complete the protocol in this scenario has a quadratic relationship, as explained in Chapter 6, with the number of mobile nodes involved. The second limitation is the limitation of the pre-handoff and handoff protocols. If handoffs are to happen in succession, i.e. a roaming node never stops moving, there is a limit, depending on the speed of the traveling node, of how much information he or she can process prior to the next handoff. We have demonstrated that the faster the speed, the less information can be processed.

Even though we consider this as a restraint, the results show that the amount of information that can be processed by the roaming node should still be adequate for the next handoff to be executed successfully.

It is also shown in Chapter 6 that both authentication and handoff protocols have satisfied the design requirements and can overcome the problems that the existing protocols have. For the authentication protocol, we have achieved mutual authentication, pairwise key establishment and (with reference to the performance and results) protocol efficiency. For the handoff protocol, we have achieved mutual authentication, make-before-break, pairwise key establishment and (with reference to the performance and results) protocol efficiency. It can also be seen that both protocols do not have to rely upon any extra third party in order to accomplish their tasks. Any node in the network can carry out the authentication protocol and the handoff protocol at any time, provided that they hold enough necessary information.

From the evidence that we have seen and presented, including the security analyses and the performances of the protocols, we claim that we have designed and developed an authentication protocol and a handoff protocol together with a pre-handoff protocol, which are suited for private local pure mobile ad hoc networks.

How will our authentication and handoff protocols be useful in the real world? In which situation(s) might the protocols be employed? First of all, our authentication protocol makes sure that a mobile ad hoc network is set up securely, and by the mobile nodes who are authorised to be part of the network only. The authentication protocol also ensures that only the authorised parties are allowed to join the network after it has been set up. Secondly, our handoff protocol ensures that the mobile nodes in the ad hoc network are able to roam securely within the network without losing the connection. Having a secure mobile ad hoc network is important and useful in the situations where there is a need of an ad hoc network being set up quickly, and where mobile nodes are required to roam around the network securely and efficiently. Such situations may include military operations, emergency rescue operations and meetings or conferences.

7.2 Future work

In this section, we discuss possible directions for future work. Specifically, we discuss how a new group key could be constructed using the existing functions in our protocols as basic functions. Next, we look at how our handoff protocol could be extended in order for it to help mobile nodes roam between different domains. Also, we explain how our pre-handoff protocol could be a basis for detecting adversaries within a mobile ad hoc network.

If and when a mobile ad hoc network becomes larger and turns into a multi-domain network, it may be necessary to create a new unique group key for each domain. One technique that could be applied to establish a group key is Cliques [STW97]. Why might this protocol be ideal for extending our protocols? The reason is that Cliques bases the computation of a new group key on Diffie-Hellman, the same method that we have employed for our pairwise key establishment. This means that our protocols have already provided the basic functions for the calculation of new group keys. Other group key generation techniques can be found in [YTCC02].

Secondly, one of the areas that this thesis focuses on is micro-mobility management in mobile ad hoc networks or movement of mobile nodes within one ad hoc network domain. It would be interesting to further our work by extending the handoff protocol, so that inter-domain handoff would be possible. We have already seen in Chapter 3 that there are many handoff protocols which assist mobile nodes in moving between domains. However, none of these existing protocols are specifically designed to work within an ad hoc environment. The solution to this macro-mobility management problem could lie in the concept of a chain of trusts which is mentioned in [OS05] and the notion of indirect trusts which is explained in [YKB93].

Another security aspect that could be extended or integrated into our system is how to detect adversaries. In more detail, apart from the authentication and secure handoff protocols, which could act as a line of defence, one may want to detect adversaries within an ad hoc network. Some efforts have been put in in order to address this problem, including [ZL00, MGLB00]. One way to detect adversaries is to see whether or not the mobile nodes in the network behave as they are supposed to. The Byzantine Generals problem [LSP82] is a possible solution to mitigate this problem. Our pre-handoff stage provides an opportunity to integrate this technique to help detect potential adversaries.

During the pre-handoff stage, a mobile node asks his multipoint relays (MPRs) to pass his security information to his two-hop neighbours. The asking node could see if the MPRs actually pass the information or not. This is possible due to the nature of wireless networks. The asking node and the MPRs are within each other's radius, therefore, if and when the MPRs send the security information (albeit not destined to the asking node), the asking node should be able to detect the signals (even though he might not be able to make any sense of what is actually sent). If the asking node cannot "hear" any signals from the MPRs, he at least knows that no information has been passed to his two-hop neighbours. Based on this, we could have a system, where each mobile node keeps a "score" for each of his MPRs. If an MPR misbehaves, the score will be deducted. The mobile nodes could then consult one another about that particular node. They could then "vote" to see whether or not that MPR is an adversary. This could be the first step towards detecting adversaries within a mobile ad hoc network.

Summary

In this thesis, we have shown a progress towards producing a secure, correct and efficient authentication and key establishment protocol for private local mobile ad hoc networks together with a secure, correct and efficient handoff protocol which is suitable for the movement of mobile nodes within one domain, i.e. micro-mobility.

Bibliography

[Aba99] Martín Abadi. Security protocols and specifications. In *Foundations of Software Science and Computation Structures: Second International Conference, FOSSACS '99*, volume 1578, pages 1–13. Springer-Verlag, Berlin Germany, 1999.

[ABABD03] Fahd Al-Bin-Ali, Prasad Boddupalli, and Nigel Davies. An inter-access point handoff mechanism for wireless network management: The sabino system. In *International Conference on Wireless Networks*, pages 225–230, 2003.

[AG00] N. Asokan and Philip Ginzboorg. Key-agreement in ad-hoc networks. *Computer Communications*, 23(17):1627–1637, 2000. Expanded version of a talk given at the Nordsec '99 workshop in Kista, Sweden, November 1999.

[AN94] Martín Abadi and Roger Needham. Prudent engineering practice for cryptographic protocols. Technical Report 125, Digital, Systems Research Center, 130 Lytton Avenue, Palo Alto, California 94301, June 1, 1994.

[ARE$^+$05] Nidal Aboudagga, Mohamed Tamer Refaei, Mohamed Eltoweissy, Luiz A. DaSilva, and Jean-Jacques Quisquater. Authentication protocols for ad hoc networks: taxonomy and research issues. In *Q2SWinet '05: Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*, pages 96–104. ACM Press, New York, NY, USA, 2005. ISBN 1-59593-241-0.

[AST98] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik. Authenticated group key agreement and friends. In *Proceedings of the 5th ACM Conference on Computer and Communications Security*, pages 17–26. ACM Press, San Francisco, California, November 1998.

[AWD04] Mehran Abolhasan, Tadeusz Wysocki, and Eryk Dutkiewicz. A review of routing protocols for mobile ad hoc networks. *Ad Hoc Networks*, 2(1):1–22, January 2004.

[BAN90] Michael Burrows, Martín Abadi, and Roger Needham. A logic of authentication. Technical Report 39, DEC System Reearch Center, February 1990. Revised version.

[BB03] Joseph Binder and Hans-Peter Bischof. Zero knowledge proofs of identity for ad hoc wireless networks, an in-depth study, January 30, 2003.

[BCK98a] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 419–428. Dallas, Texas, USA, 1998.

[BCK98b] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (full version), March 1998. Available at `http://www-cse.ucsd.edu/users/mihir` and at the Theory of Cryptography Library, `http://theory.lcs.mit.edu/∼tcryptol`.

[BD95] Mike Burmester and Yvo Desmedt. A secure and efficient conference key distribution system. In *Proc of Eurocrypt' 94, vol. 950 of LNCS*, pages 275–286. Springer, 1995.

[BDLP88] Jørgen Brandt, Ivan Damgård, Peter Landrock, and Torben P. Pedersen. Zero-knowledge authentication scheme with secret key exchange. In *CRYPTO*, pages 583–588, 1988.

[Bet88] T. Beth. Efficent zero-knowledge identification scheme for smart cards. In C. G. Gnther, editor, *Advances in Cryptology: Proceedings* *of Proceedings of Eurocrypt '88*, volume 330 of *Lecture Notes in Computer Science*, pages 77–89. Springer-Verlag, 1988.

[BF01] Dan Boneh and Matthew Franklin. Identity-based encryption from the Weil Pairing. *Lecture Notes in Computer Science*, 2139:213–229, 2001. This is the full version of an earlier published in Crypto '2001.

[BHE$^+$04] M. S. Bargh, R. J. Hulsebosch, E. H. Eertink, A. Prasad, H. Wang, and P. Schoo. Fast authentication methods for handovers between ieee 802.11 wireless lans. In *WMASH'04: Proceedings of the 2nd ACM international workshop on wireless mobile applications and services on WLAN hotspots*, pages 51–60. ACM Press, New York, NY, USA, 2004.

[BSSW02] Dirk Balfanz, D. K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *In Symposium on Network and Distributed Systems Security (NDSS '02)*, 2002.

[BT01] Jim Binkley and William Trost. Authenticated ad hoc routing at the link layer for mobile systems. *Wireless Networks*, 7(2):139–145, 2001.

[CCG97] W. Liu C. Chiang, H. Wu and M. Gerla. Routing in clustered multihop, mobile wireless networks. In *Proc. IEEE SICON'97*, pages 197–211, April 1997.

[Che97] Steven Cheung. Efficient message authentication scheme for link state routing. In *Proc.* 13th *Annual Computer Security Applications Conference*. San Diego, Calif, December 8–12, 1997.

[CJ03] T. Clausen and P. Jacquet. Optimized link state routing protocol (olsr). RFC 3626, IETF, October 2003.

[CK01] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *Proceedings of the International Conference on the Theory and Application of*

Cryptographic Techniques: Advances in Cryptology, pages 453–474. Springer-Verlag, 2001.

[CM99] S. Corson and J. Macker. Mobile ad hoc networking (manet). RFC 2501, IETF, January 1999.

[CP96] Ramon Caceres and Venkata N. Padmanabhan. Fast and scalable handoffs for wireless internetworks. In *Proceedings of the 2nd annual international conference on Mobile computing and networking*, pages 56–66. ACM Press, 1996.

[Den93] Dorothy E. Denning. A new paradigm for trusted systems. In *NSPW '92-93: Proceedings on the 1992-1993 workshop on New security paradigms*, pages 36–41. ACM Press, New York, NY, USA, 1993. ISBN 0-8186-5430-9.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[DLRS02] B. Dahill, B. N. Levine, E. Royer, and C. Shields. A secure routing protocol for ad hoc networks. In *Proc. of International Conference on Network Protocols (ICNP)*, pages 78–87, 2002.

[DRWT97] R. Dube, C. Rais, K. Wang, and S. Tripathi. Signal stability based adaptive routing (ssa) for ad hoc mobile networks, 1997.

[DvOW92] W. Diffie, P. van Oorschot, and M. Wiener. Authentication and authenticated key exchange. *Designs, Codes and Cryptography*, 2: 107–125, 1992.

[FFS87] U. Fiege, A. Fiat, and A. Shamir. Zero knowledge proofs of identity. In *STOC '87: Proceedings of the nineteenth annual ACM conference on Theory of computing*, pages 210–217. ACM Press, 1987. ISBN 0-89791-221-7.

[FJL00] Magnus Frodigh, Per Johansson, and Peter Larsson. Wireless ad hoc networking - the art of networking without a network. *Ericsson Review*, Year 2000(4):248–263, 2000.

[Gem97] Peter S. Gemmell. An introduction to threshold cryptography. *CryptoBytes*, 2(3):7–12, Winter 1997.

[GNY90] Li Gong, Roger Needham, and Raphael Yahalom. Reasoning about belief in cryptographic protocols. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 234–248. IEEE Computer Society, Technical Committee on Security and Privacy, IEEE Computer Society Press, Oakland, CA, May 1990.

[GSG99] Stefanos Gritzalis, Diomidis Spinellis, and Panagiotis Georgiadis. Security protocols over open networks and distributed systems: Formal methods for their analysis, design, and verification. *Computer Communications*, 22(8):695–707, May 1999.

[HBC01] J. P. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad hoc networks. In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*. Long Beach, CA, USA, October 2001.

[Hie00] Maarit Hietalahti. Key establishment in ad-hoc networks, 2000.

[HP99] Z. J. Haas and R. Pearlman. Zone routing protocol for ad-hoc networks. Internet-Draft Version 02, IETF, 1999.

[JM96] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Imielinski and Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[JNL99a] M Joa-Ng and I.-T. Lu. A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks*, pages 1415–1425, August 1999.

[JNL99b] M. Joa-Ng and I-Tai Lu. A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 17(3):1415–1425, August 1999.

[KBC97] H. Krawczyk, M. Bellare, and R. Canetti. HMAC: Keyed-Hashing for Message Authentication. RFC 2104, IETF, February 1997.

[KKA03] Aram Khalili, Jonathan Katz, and William A. Arbaugh. Toward secure key distribution in truly ad-hoc networks. In *Proceedings of the IEEE Workshop on Security and Assurance in Ad hoc Networks, in conjunction with the 2003 International Symposium on Applications and the Internet*. Orlando, FL, January 28, 2003.

[KP01] Rajeev Koodli and Charles E. Perkins. Fast handovers and context transfers in mobile networks. *SIGCOMM Comput. Commun. Rev.*, 31(5):37–47, 2001.

[Kra01] Hugo Krawczyk. The order of encryption and authentication for protecting communications (or: How secure is ssl?). In *Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology*, pages 310–331. Springer-Verlag, 2001.

[KZ04] Tim Kindberg and Kan Zhang. Secure spontaneous interactions in ubiquitous computing, February 2004. Talk given at university of Bristol, UK.

[KZL$^+$01] Jiejun Kong, Petros Zerfos, Haiyun Luo, Songwu Lu, and Lixia Zhang. Providing robust and ubiquitous security support for mobile ad hoc networks. In *ICNP '01: Proceedings of the Ninth International Conference on Network Protocols (ICNP'01)*, page 251. IEEE Computer Society, Washington, DC, USA, 2001.

[LABW92] Butler Lampson, Martin Abadi, Michael Burrows, and Edward Wobber. Authentication in distributed systems: Theory and practice. *ACM Trans. Comput. Syst.*, 10(4):265–310, 1992. ISSN 0734-2071.

[LL00] Haiyun Luo and Songwu Lu. Ubiquitous and robust authentication services for ad hoc wireless networks. Research Report TR-200030, Dept. of Computer Science, UCLA, October 2000.

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, 1982. ISSN 0164-0925.

[MF] S. McCanne and S. Floyd. The network simulator - ns2. www.isi.edu/nsnam/ns.

[MGLA96] Shree Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Mob. Netw. Appl.*, 1(2):183–197, 1996. ISSN 1383-469X.

[MGLB00] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.

[MSA03] Arunesh Mishra, Minho Shin, and William Arbaugh. An empirical analysis of the ieee 802.11 mac layer handoff process. *ACM SIGCOMM Computer Communication Review*, 33(2):93–102, 2003.

[MSNN94] Anish Mathuria, Reihaneh Safavi-Naini, and Pater Nickolas. Some remarks on the logic of gong, needham and yahalom. In *Proceedings of the International Computer Symposium 1994*, volume 1, pages 303–308. NCTU, Hsinchu, Taiwan, December 12–15, 1994.

[MvOV96] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography.* CRC Press LLC, 1996.

[New04] Calvin Newport. Simulating mobile ad hoc networks: a quantitative evaluation of common MANET simulation models. Technical Report TR2004-504, Dartmouth College, Computer Science, Hanover, NH, June 2004.

[NLB00] N. Nikaeim, H. Laboid, and C. Bonnet. Distributed dynamic routing (ddr) algorithm for mobile ad hoc networks. In *Proceedings of the MobiHOC 2000: First Annual Workshop on Mobile Ad Hoc Networking and Computing*, 2000.

[NS00] Dae Hun Nyang and Joo Seok Song. Knowledge-proof based versatile smart card verification protocol. *SIGCOMM Comput. Commun. Rev.*, 30(3):39–44, 2000. ISSN 0146-4833.

[Odl87] Andrew M. Odlyzko, editor. *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, volume 263 of *Lecture Notes in Computer Science*. Springer, 1987.

[OS05] K. Ono and H. Schulzrinne. Trust path discovery. Internet-Draft, IETF, July 2005.

[PB94] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[PC97] Vincent D. Park and M. Scott Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *INFOCOM (3)*, pages 1405–1413, 1997.

[PC02a] Sangheon Pack and Yanghee Choi. Fast inter-ap handoff using predictive-authentication scheme in a public wireless lan. In *Proceedings of IEEE Networks 2002 (Joint ICN 2002 and ICWLHN 2002)*, August 2002.

[PC02b] Sangheon Pack and Yanghee Choi. Pre-authenticated fast handoff in a public wireless lan based on ieee 802.1x model. In *Proceedings of the IFIP TC6/WG6.8 Working Conference on Personal Wireless Communications*, pages 175–182. Kluwer, B.V., 2002.

[PC03] Eun Kyoung Paik and Yanghee Choi. Prediction-based fast handoff for mobile wlans. In *10th International Conference on Telecommunications (ICT 2003)*, pages 748–753, February 2003.

[Per96] Charles Perkins. IP Mobility Support. RFC 2002, IETF, October 1996.

[PGC00] Guanyu Pei, Mario Gerla, and Tsu-Wei Chen. Fisheye state routing in mobile ad hoc networks. In *ICDCS Workshop on Wireless Networks and Mobile Computing*, pages D71–D78, 2000.

[PH02] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure routing for mobile ad hoc networks. In *SCS Communication Networks and*

Distributed Systems Modeling and Simulation Conference (CNDS 2002). San Antonio, TX, January 27–31, 2002.

[PH03] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure link state routing for mobile ad hoc networks. In *SAINT-W '03: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT'03 Workshops)*, page 379. IEEE Computer Society, Washington, DC, USA, 2003. ISBN 0-7695-1873-7.

[PM04] Asad Amir Pirzada and Chris McDonald. Establishing trust in pure ad-hoc networks. In *CRPIT '04: Proceedings of the 27th conference on Australasian computer science*, pages 47–54. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 2004.

[PR99] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of 2nd IEEE Workshop Mobile Computer Systems and Applications*, pages 90–100, February 1999.

[Pro03] The VINT Project. The ns manual (formerly ns notes and documentation), December 2003. A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and XEROX PARC.

[QGAB89] Jean-Jacques Quisquater, Louis Guillou, Marie Annick, and Tom Berson. How to explain zero-knowledge protocols to your children. In *Proceedings on Advances in cryptology*, pages 628–631. Springer-Verlag New York, Inc., 1989.

[QVL00] Amir Qayyum, Laurent Viennot, and Anis Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. In *35th Hawaii International Conference on System Sciences (HICSS-35)*, March 2000.

[RB01] Pierre Reinbold and Olivier Bonaventure. A comparison of ip mobility protocols. Technical report, University of Namur, Infonet Group, 2001.

[Rep03] BWCS Report. Wisp roaming - single subscription, global reach, May 2003.

[RLMD00] C. Rigney, S. Willens Livingston, A. Rubens Merit, and W. Simpson Daydreamer. Remote authentication dial in user service (RADIUS). RFC 2865, IETF, June 2000.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems, February 1978.

[RT99] Elizabeth M. Royer and Chai-Keong Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55, 1999.

[SA99] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In B. Christianson, B. Crispo, and M. Roe, editors, *Security Protocols, 7th International Workshop Proceedings*, Lecture Notes in Computer Science, 1999, pages 172–194. Springer-Verlag Berlin Heidelberg, 1999, 1999.

[Sal02] A O Salako. Authentication in ad hac networking. In *Proceedings of London Communications Symposium 2002*, 2002.

[SGLA96] Bradley R. Smith and J. J. Garcia-Luna-Aceves. Securing the border gateway routing protocol. In *Global Internet'96*, November 20–21, 1996.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology: Proceedings of CRYPTO 84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1985, 19–22 August 1984.

[Sta01] Frank Stajano. The resurrecting duckling - what next? In *Revised Papers from the 8th International Workshop on Security Protocols*, pages 204–214. Springer-Verlag, London, UK, 2001. ISBN 3-540-42566-7.

[Sta03] Tyron Stading. Secure communication in a distributed system using identity based encryption. In *Proceedings of Cluster Computing and the Grid*, pages 414–420, 2003.

[STW97] Michael Steiner, Gene Tsudik, and Michael Waidner. CLIQUES: A new approach to group key agreement. Research Report RZ 2984 (# 93030), IBM Research, December 22, 1997.

[SZ98] Q. Shi and N. Zhang. An effective model for composition of secure systems. *The journal of Systems and Software*, 43(3):233–244, November 1998. ISSN 0164-1212.

[TGLN05] Christian Tschudin, Per Gunningberg, Henrik Lundgren, and Erik Nordstrom. Lessons from experimental MANET research. *Ad Hoc Networks Journal, Special Issue on "Ad Hoc Networking for Pervasive Systems"*, 3(2):221–233, March 2005.

[TLP99] Cheng Lin Tan, Kin Mun Lye, and Stephen Pink. A fast handoff scheme for wireless networks. In *Proceedings of the 2nd ACM international workshop on Wireless mobile multimedia*, pages 83–90. ACM Press, 1999.

[Toh97] Chai-Keong Toh. Associativity-based routing for ad hoc mobile networks. *Wirel. Pers. Commun.*, 4(2):103–139, 1997. ISSN 0929-6212.

[VA00] Lakshmi Venkatraman and Dharma P. Agrawal. A novel authentication scheme for ad hoc networks. In *WCNC2000:IEEE Wireless Communications and Networking Conference 2000*, September 23–28, 2000.

[WS01] Seung-Chul M. Woo and Suresh Singh. Scalable routing protocol for ad hoc networks. *Wirel. Netw.*, 7(5):513–529, 2001. ISSN 1022-0038.

[WYOP94] William A. Wulf, Alec Yasinsac, Katie S. Oliver, and Ramesh Peri. A technique for remote authentication. In *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, pages 159–164, 1994.

[YKB93] R. Yahalom, B. Klein, and T. Beth. Trust relationships in secure systems-a distributed authentication perspective. In *SP '93: Proceedings of the 1993 IEEE Symposium on Security and Privacy*, pages 150–164. IEEE Computer Society, Washington, DC, USA, 1993.

[YTCC02] Alec Yasinsac, Vikram Thakur, Stephen Carter, and Ilkay Cubukcu. A family of protocols for group key generation in ad hoc networks. In *the IASTED International Conference on Communications and Computer Networks (CCN02)*, November 2002.

[ZH99] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.

[ZL00] Yongguang Zhang and Wenke Lee. Intrusion detection in wireless ad-hoc networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 275–283. ACM Press, New York, NY, USA, 2000. ISBN 1-58113-197-6.

Appendix A

GNY

In this appendix, we list the notations for the GNY logic and the logical postulates (or rules) that are used for the analysis and proof for correctness. The GNY protocol was developed by Gong, Roger and Yahalom as an analysis method for cryptographic protocols. The contents in this appendix are extracted from [GNY90].

A.1 GNY notations

Let P and Q be principals. The following are the basic notations used in the GNY protocol.

$P \triangleleft X$: P *is told* formula X. P receives X, possibly after performing some computation such as decryption. That is, a formula being told can be the message itself, as well as any computable content of that message.

$P \ni X$: P *possesses*, or is capable of possessing, formula X. At a particular stage of a run, this includes all the formulae that P has been told, all the formulae he started the session with, and all the ones he has generated in that run. In addition P possesses, or is capable of possessing, everything that is computable from the formulae he already possesses.

$P \mid\sim X$: P *once conveyed* formula X. X can be a message itself or some content computable from such a message, i.e. a formula can be conveyed implicitly.

$P \mid\equiv \sharp(X)$: P *believes*, or is entitled to believe, that formula X is *fresh*. That is, X has not been used for the same purpose at any time before the current run of the protocol.

$P \mid\equiv \phi(X)$: P *believes*, or is entitled to believe, that formula X is *recognisable*. That is, P would recognise X is P has certain expectations about the contents of X before actually receiving X. P may recognise a particular value (e.g. his own identifier), a particular structure (e.g. the format of a timestamp), or a particular form of redundancy.

$P \mid\equiv P \overset{S}{\leftrightarrow} Q$: P *believes*, or is entitled to believe, that S is a suitable *secret* for P and Q. S will never be discovered by any principal except P, Q. This notation is symmetrical: $Q \overset{S}{\leftrightarrow} P$ and $P \overset{S}{\leftrightarrow} Q$ can be used interchangeably.

$P \mid\equiv \overset{+K}{\mapsto} Q$: P *believes*, or is entitled to believe, that $+K$ is a suitable *public key* for Q. The matching secret key $-K$ will never be discovered by any principal except Q.

$P \lhd *X$: P is told a formula which he did not convey previously in the current run. That is, X can be regarded as a *not-originated-here* formula.

Let C be a statement.
$P \mid\equiv C$: P believes, or P would be entitled to believe, that statement C holds.

A.2 Logical postulates

In this section, we list the logical postulates that are used in the analysis and proof of correctness in Chapter 5. The description of each of the rule is also given.

A.2.1 Being-told rules

T1: $\dfrac{P \lhd *X}{P \lhd X}$

Being told a "not-originated-here" formula is a special case of being told a formula

T3: $\dfrac{P \lhd \{X\}_K, P \ni K}{P \lhd X}$

If a principal is told a formula encrypted with a key he possesses then he is considered to have also been told the decrypted contents of that formula.

T4: $\dfrac{P \lhd \{X\}_{+K}, P \ni -K}{P \lhd X}$

If a principal is told a formula encrypted with a public key and he possesses the corresponding private key then he is considered to have also been told the decrypted contents of that formula.

A.2.2 Possession rules

P1: $\dfrac{P \lhd X}{P \ni X}$

A principal is capable of possessing anything he is told.

P4: $\dfrac{P \ni X}{P \ni H(X)}$

If a principal possesses a formula then he is capable of possessing a one-way computationally feasible function of that formula.

A.2.3 Freshness rule

F1: $\dfrac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X, Y)}$

If P believes a formula X is fresh, then he is entitled to believe that any formula of which X is a component is fresh.

A.2.4 Recognisability rule

R1: $\dfrac{P \mid\equiv \phi(X)}{P \mid\equiv \phi(X, Y)}$

If P believes a formula X is recognisable, then he is entitled to believe that any formula of which X is a component is recognisable.

A.2.5 Message interpretation rules

I1: $\dfrac{P \lhd *\{X\}_K, P \ni K, P \mid\equiv P \overset{K}{\leftrightarrow} Q, P \mid\equiv \phi(X), P \mid\equiv \sharp(X, K)}{P \mid\equiv Q \mid\sim X, P \mid\equiv Q \mid\sim \{X\}_K, P \mid\equiv Q \ni K}$

Suppose that for principal P all of the following conditions hold: (1) P receives a formula consisting of a X encrypted with key K and marked with a not-originated-here mark; (2) P possesses K; (3) P believes K is a suitable secret for himself and Q; (4) P believes formula X is recognisable; (5) P believes that K is fresh or that X is fresh.

Then P is entitled to believe that (1) Q once conveyed X; (2) Q once conveyed the formula X encrypted with K; (3) Q possesses K.

I4: $\dfrac{P \lhd \{X\}_{-K}, P \ni +K, P \mid\equiv \overset{+K}{\mapsto} Q, P \mid\equiv \phi(X)}{P \mid\equiv Q \mid\sim X, P \mid\equiv Q \mid\sim \{X\}_{-K}}$

Suppose for principal P, all of the following conditions hold: (1) P receives a formula consisting of X encrypted with a private key; (2) P possesses the corresponding public key; (3) P believes that public key is Q's; (4) P believes X is recognisable.

Then P is entitled to believe that (1) Q once conveyed the formula X; (2) Q once conveyed the formula consisting of X encrypted with the private key.

A.2.6 Jurisdiction rules

J1: $\dfrac{P \mid\equiv Q \mid\Longrightarrow C, P \mid\equiv Q \mid\equiv C}{P \mid\equiv C}$

If P believes that Q is an authority on some statement C and that Q believes in C, then P ought to believe in C as well.

$$\textbf{J2}\text{:}\ \frac{P \mid\equiv Q \mid\Longrightarrow Q \mid\equiv *, P \mid\equiv Q \mid\sim (X \leadsto C), P \mid\equiv \sharp(X)}{P \mid\equiv Q \mid\equiv C}$$

If P believes that Q is honest and competent, and P receives a message $X \leadsto C$ which he believes Q conveyed, then P ought to believe that Q really believes C.

Appendix B

Network Layouts

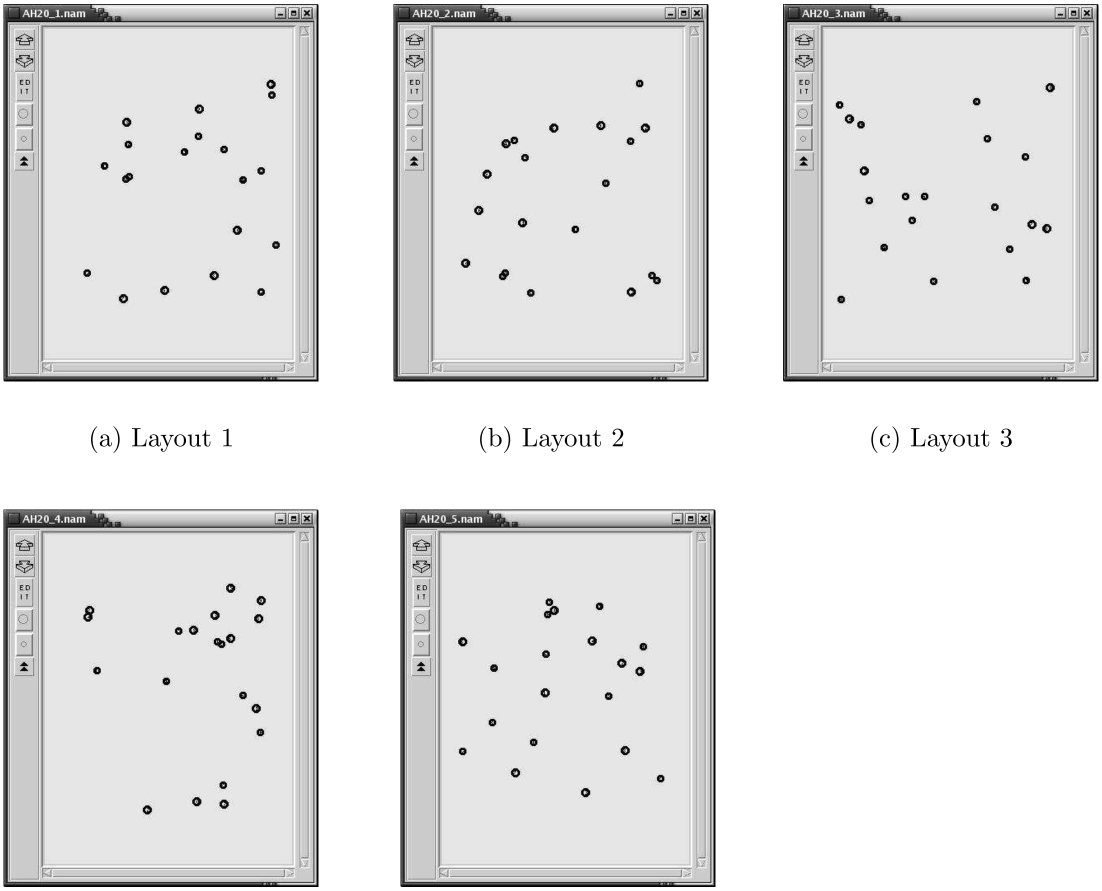
We provide the diagrams showing the layouts of the network that are used in the simulations described in Section 5.3. In the first section, we provide five different layouts for twenty mobile nodes. Second section gives five layouts for fifty mobile nodes. In the third section, we show five layouts for one hundred nodes. The network layouts for two hundred and two hundred and fifty nodes are shown in Section B.5 and Section B.6 respectively.

The positions of the mobile nodes are randomly chosen by running a program called `setdest`, which can be found in `ns-2.27/indep-utils/cmu-scen-gen/setdest/`. `setdest` allows *ns2* users to specify the number of mobile nodes and the area in which they will be placed.

Note that having different layouts for each set of mobile nodes does not affect the performances or the ways the protocols behave.

B.1 20 nodes

The following figures show how twenty mobile nodes are laid out each time the simulation is run.



(a) Layout 1

(b) Layout 2

(c) Layout 3

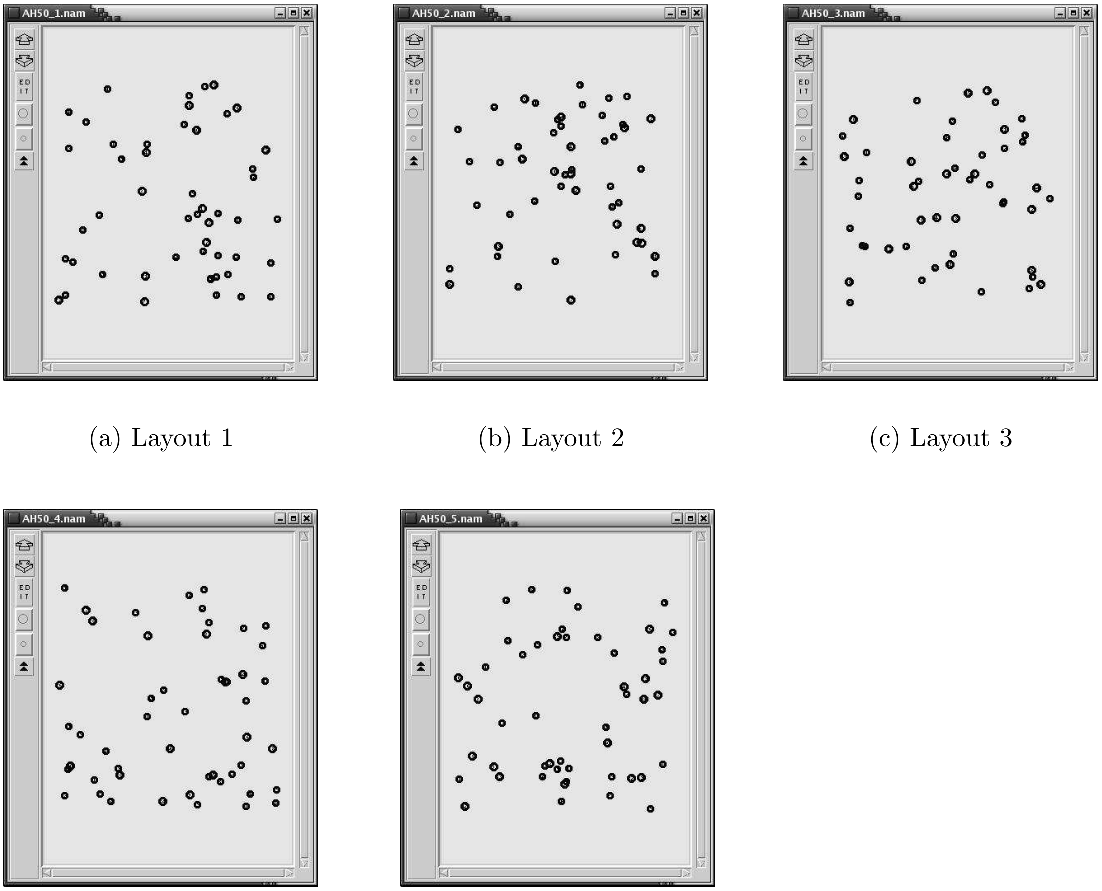
(d) Layout 4

(e) Layout 5

Figure B-1: Network layouts for 20 mobile nodes

B.2 50 nodes

The following figures show how fifty mobile nodes are laid out each time the simulation is run.



(a) Layout 1

(b) Layout 2

(c) Layout 3

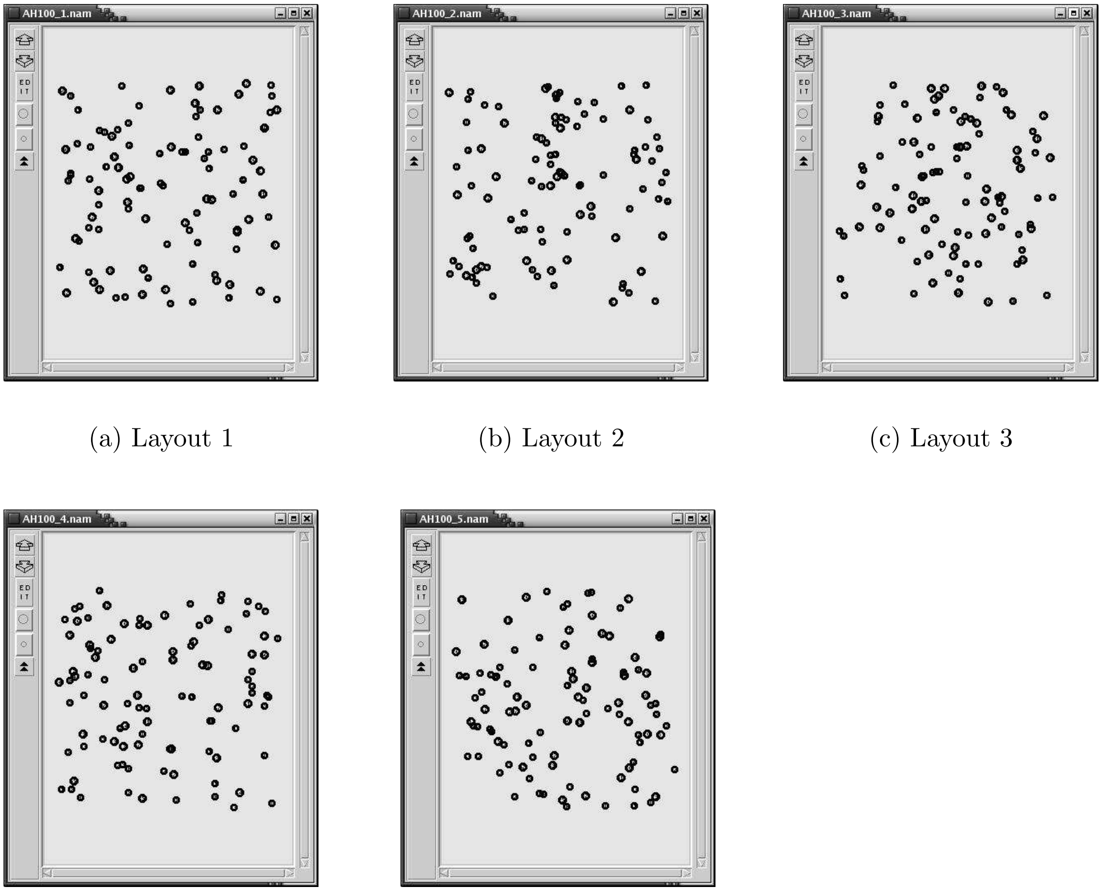
(d) Layout 4

(e) Layout 5

Figure B-2: Network layouts for 50 mobile nodes

B.3 100 nodes

The following figures depicts the layout of one hundred mobile nodes each time the simulation is run.



(a) Layout 1

(b) Layout 2

(c) Layout 3

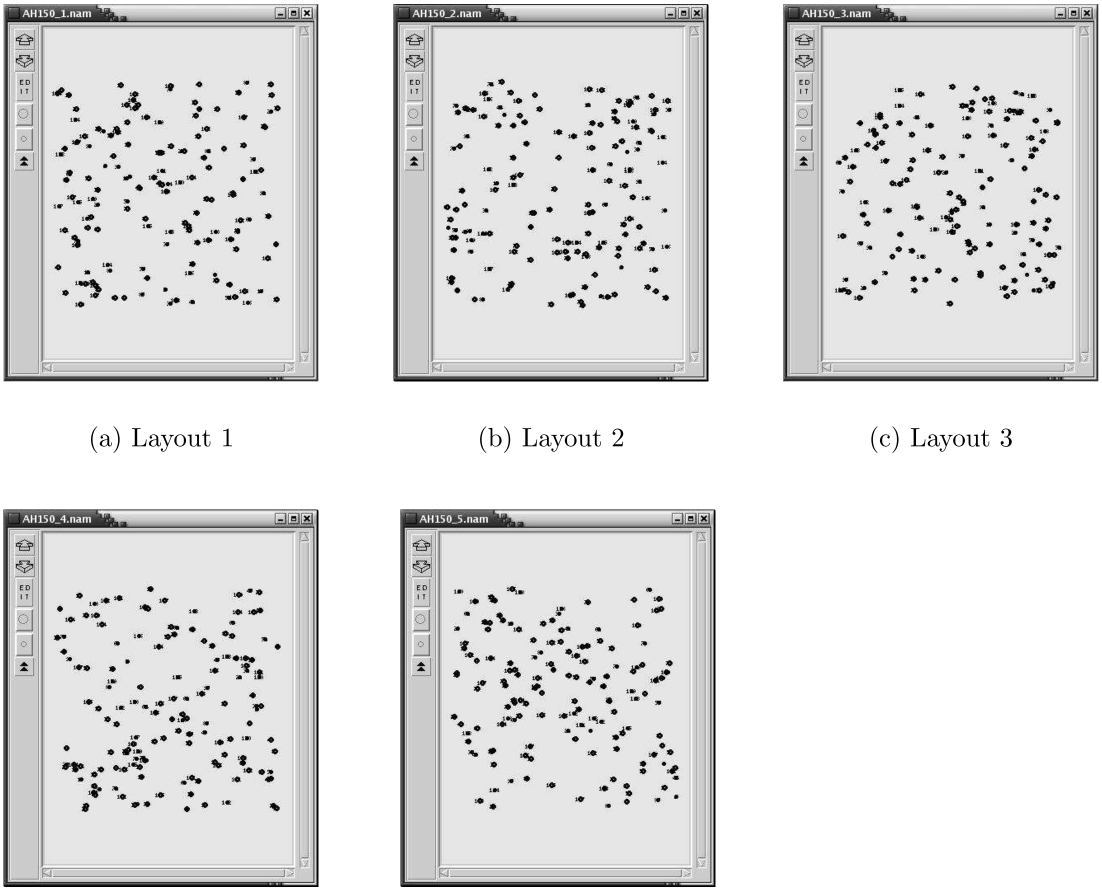
(d) Layout 4

(e) Layout 5

Figure B-3: Network layouts for 100 mobile nodes

B.4 150 nodes

The following figures depicts the layout of one hundred and fifty mobile nodes each time the simulation is run.



(a) Layout 1

(b) Layout 2

(c) Layout 3

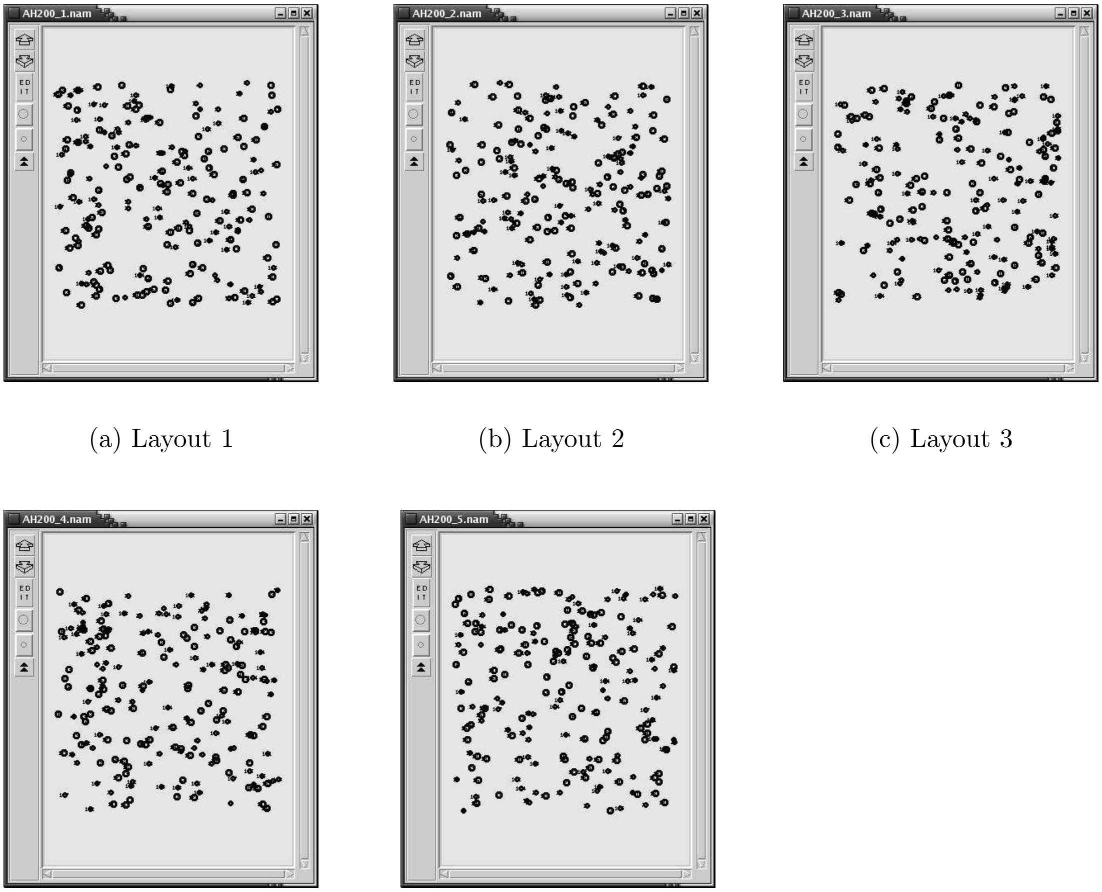
(d) Layout 4

(e) Layout 5

Figure B-4: Network layouts for 150 mobile nodes

B.5 200 nodes

Below are the diagrams of the network layouts of two hundred mobile nodes.



(a) Layout 1

(b) Layout 2

(c) Layout 3

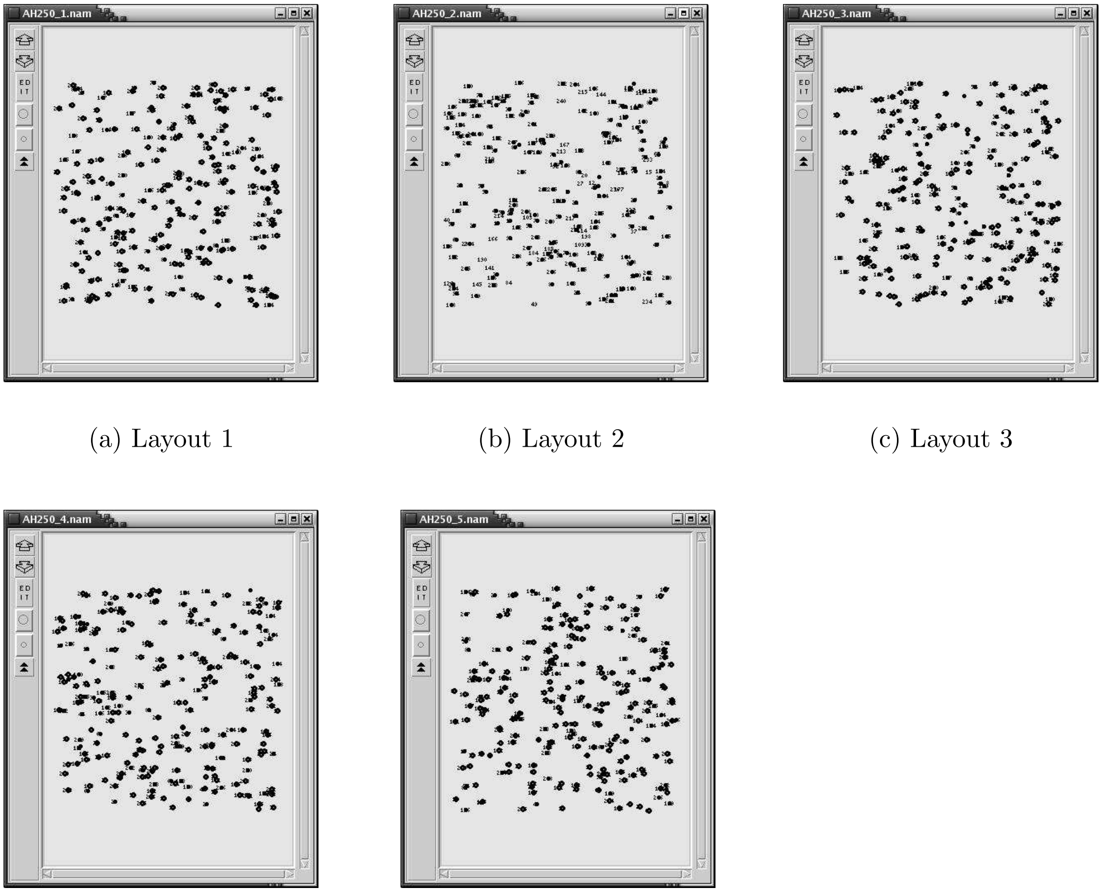
(d) Layout 4

(e) Layout 5

Figure B-5: Network layouts for 200 mobile nodes

B.6 250 nodes

Two hundred and fifty mobile nodes are placed in the network as follows.



(a) Layout 1

(b) Layout 2

(c) Layout 3

(d) Layout 4

(e) Layout 5

Figure B-6: Network layouts for 250 mobile nodes