Authentication in Mobile Ad-hoc network

Ståle Jonny Berget

Master's Thesis
Master of Science in Information Security
30 ECTS
Department of Computer Science and Media Technology
Gjøvik University College, 2006

Institutt for
informatikk og medieteknikk
Høgskolen i Gjøvik
Postboks 191
2802 Gjøvik

Department of Computer Science
and Media Technology
Gjøvik University College
Box 191
N-2802 Gjøvik
Norway

Abstract in English

A proper authentication protocol is necessary to establish appropriate integrity in a Mobile Ad-hoc network (MANET). The big issue in authentication is how to assure that only legitimate nodes are part of the network. The MANET can be divided in two groups: 1) MANET with one single authority and 2) MANET without a single authority. This thesis proposes a new authentication protocol for MANET with one single authority as in a rescue operation scenario. The protocol is divided in two part the first is a protocol that authenticate message at each hop between source and destination node, and the other part take care of end to end authentication. The protocol protect against DoS attack, and the destination node is able to detect tampered (change, manipulated or extended) messages which may be caused by a wormhole or insider attack. To design this protocol the thesis considers different threat that must be expected in this kind of network, and a description of the scenario. The protocol is based on late disclosure of key, proposed by Lesli Lamport, and use hash, HMAC and public/private key function. The protocol requires that all nodes have synchronically clock that may be supported by a GPS receiver.

Keyword: Authentication, Mobile Ad-hoc Network, MANET.

Abstract in Norwegian

En egnet autentiserings protokoll er nødvendig for å oppnå tilfredsstillende integritet i et mobilt ad-hoc nettverk (MANET). En av hovedutfordringene er å sikre at kun legale noder er medlemmer (brukere) av nettverket, og at autentisering av noder er sikker og energi effektiv. Mobile ad-hoc nettverk kan deles inn i to grupper: 1) MANET med en autoritet og 2) MANET uten en autoritet. Masteroppgaven foreslår en ny autentisering-sprotokoll med en tiltrodd tredje part (TTP), som benyttes til å ustede sertifikater med tilhørende private/offentlike nøkler før en node blir deployert. Protokollen er to delt; hvor den ene håndterer autentisering for hvert hopp i nettverket mellom kilde og mottaker, autentisering av nye noder og når noder beveger seg rundt i nettverket. Den andre håndtere ende-til-ende autentisering. Protokollen er hovedsaklig basert på bruk av hash kjeder, som en erstatning for digitale signatur. Verdiene fra hash kjedene, benyttes som nøkkeler i meldingsautentiseringskoden. Den første hash nøkkelen er signert ved bruk av offentlig/private nøkler. For å unngå tjenestenekt angrep (DoS-attack), må noden som initierer autentisering bevise at den virkelig ønker å fullføre authentiseringen. Ende-til-ende autentisering sikrer at en kompromittert node i nettet og "wormhole attack", ikke kan manipulere meldinger som sendes gjennom nettet, uten at mottaker er i stand til å detektere det. Den ny authetiserings protokollen er basert på grunnlag fra "TESLA" protokollen og Lesseli Lamports "late disclosure of key".

Preface

This master thesis is a part of the fulfilment of my Master's degree in Information Security at Gjøvik University College, where I have been a student from autumn 2003 to autumn 2006. Before that I received the Engineer degree in Telematic from Gjøvik University College in 1989, and Sivilingeniør degree from Norwegian Institute of Technology (to day Norwegian University of Science and Technology) at the faculty of Electrical Engineering and Computer Science in May 1993.

Acknowledgement

I would like to thank my supervisor, Professor Chik How Tan, for his inspiration and support. He has been of great help, and given me constructive comments, ideas, advice and been patient during the time, and made it possible to finish this master thesis.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

1.1 Topic covered by this thesis

Today companies, organisations and people in the western country use different kind of information, communication technology (ICT) services which are more mobile. This equipment is used by the organisation to be more efficient and timely. For this organisation most company the ICT services is depending on a wired communication network or wireless network to get access to a local area network or Internet. In some situation where there are not any kind of network as in rescue operation e.g. after a Tornado or Tsunami where the network may be damaged, there may be necessary to establish a mobile ad-hoc network (MANET) to run the rescue operation with better efficient and to gain timely delivery of support.

The master thesis covers a description of the scenario where a MANET is used in a rescue operation, which may be generalised to include scenario where there is one single authority. The single authority may be any organisation, and cover a hierarchy of other organisation that is approved by the single authority. The master thesis cover authentication between nodes and the problem that is related to establish authentication of new node and node that is joining, moving or leaving the network.

1.2 Problem description

The problem in MANET is mostly related to that there is not any central management system or access to a trusted third party (TTP), which contain a repository of the identity of each legitimate node of the network. It must be assumed, that node have restricted computation power, power and memory capacity. In addition node may frequently change location or new one is entering the network. It must also be assumed, that the network will be exposed for passive and active attack from an unauthorised source. The new authentication protocol must take care of this restricted resource and threats that may be assumed.

1.3 Justification, motivation and benefits

A MANET may be useful in many situations where no fixed infrastructure is available, or as a wireless access in urban areas, to providing quick deployment and extended coverage. Without an appropriate authentication protocol, it is possible that the network may be used by user that does not follow legal principle, or is not a legitimate user (node). At this moment there are not any standards that describe a proper authentication protocol, which may be use in MANET. There are still lot of ongoing researches in this area, but according to [17] authentication and key exchange for MANET is a security problem, which still is not satisfactory solved.

1.4 Research questions

The master thesis cover following research question in a rescue operation scenario:

- Description of the scenario for the rescue operation.
- What kind of threat that may be expected for MANET in this scenario.
- Consideration on what kind of different authentication method and cryptographic algorithm, which may be appropriate against the threat and useful in a MANET.
- Design of a new and better authentication protocol, which is suited for this scenario.

1.5 Summery of Claimed Contribution

The master thesis discuses a threat model for MANET in the scenario of a rescue operation [39]. The design of the new authentication protocol cover an analyse of different authentication method, and cryptographic algorithm that satisfy the restricted capacity that must be assumed for MANET according to [16, 17, 38]. It also includes consideration on that the node is mobile and may frequently move to another location and suppose to join the network at the new location, or a new entity/node that are joining the network. The thesis discus earlier proposed authentication protocol, and what security and efficient constraints that is related to these protocols. Especial when a node or entity is moving from one location to other location. At the end it will be a consideration on how the proposed authentication protocol can be use in more general scenario with one single authority, and how it is feasible to a scenario without a single authority.

1.6 Choice of methods

The research method that has been used during the master thesis is kinds of a case study. That includes organisation of the details that best describe the scenario for a rescue operation, and what must be taken into consideration when using a MANET in this situation. To establish the threat model, it has been necessary to do a survey on what kind of threat that may be expected for this scenario. The survey is mostly based on earlier works and proposed authentication protocol, since there are not so many papers that only cover threat that may be assumed for MANET. On basis of qualitative method decide what kind of cryptographic algorithm, which may be used by node with restricted capacity mention in [16, 17, 38]. To assure that the proposed authentication protocol is compliant to the threat model, it is necessary to do a review of the protocol that considers this different threat. At the end, the proposed authentication protocol will be reviewed, to consider how it protects against threats according to the threat model.

The qualitative method that has been used to consider different cryptographic method and the new authentication protocol is based on earlier work, literature [25, 34] and paper. It has been necessary to use mathematical consideration to compare the new protocol, with earlier proposed protocols. This is also the regular method that is used to compare protocol and cryptographic methods. Typical parameter that is considered is how efficient the new protocol is, the need of computational power to complete the authentication scheme, how much memory space it require and how resistant it is against attack. Microsoft Excel and a test program is used to test the new protocol. The test program, is a C-language program according to appendix D, which is base on Multi precision Integer and Rational Arithmetic C/C++ Library (Miracl) from Shamus Software Ltd [22].

2 Definition and terminology

2.1 Definition

In this section some of notation that is use is defined.

According [28], Mobile Ad-hoc Network (MANET) and Wireless Ad hoc Sensor Network is defined as:

- MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links. Since the nodes are mobile, the network topology may change rapidly and unpredictably over time. The network is decentralized, where all network activity including discovering the topology and delivering messages must be executed by the nodes them selves, i.e., routing functionality will be incorporated into mobile nodes. The set of applications for MANETs is diverse, ranging from small, static networks that are constrained by power sources, to large-scale, mobile, highly dynamic networks.
- Wireless Ad hoc Sensor Network consists of a number of sensors spread across a geographical area. Each sensor has wireless communication capability and some level of intelligence for signal processing and networking of the data.

In this report, node, neighbour node and neighbourhood are used and are defined as:

- Node is a device that communicates or routes information within a MANET.
- A neighbour node to a node is defined as another node that is within radio coverage to this node
- Neighbourhood to a node is defined as all nodes within radio coverage to this node.

2.2 Terminology

CA	Certificate Authority
DoS	Denial of Service
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
GPS	Global Positioning System
GSM	Global System for Mobile Communications
Hash	Is a way of creating a small digital "fingerprint" from any kind of data
KDC	Key Distribution Centre
MAC	Message Authentication Codes
MANET	Mobile Ad-Hoc Network
Miracl	Multi precision Integer and Rational Arithmetic C/C++ Library
NIST	National Institute of Standards and Technology (US)
Node	Se definition
PKI	Public key infrastructure
RSA	A asymmetric encryption algorithm that was described in 1977 by Ron Rivest, Adi

	Shamir and Len Adleman at MIT; the letters RSA are the initials of their surnames
SHA	Secure Hash Algorithm
TESLA	Timed Efficient Stream Loss-tolerant Authentication
TETRA	Terrestrial Trunked Radio
TTP	Trusted Third Party
WLAN	Wireless local-area network

3 Overview of mobile ad-hoc network

3.1 Scenario

In a rescue operation, it is expected that different kind of equipment is used to support the mission. There may be vehicle, foot on ground, helicopters etc., to support the mission, it may be necessary to make it possible for different unit to communicate and share information. To support this, nods may be mounted or carried on vehicle, by person etc, or as a static node that does not move.

As the network topology in a MANET is rapidly changing, due to a node may frequently join or leave a location in the network. Other part of the network may be static, to assure that the network cover an area of interest. To obtain connection beyond line of sight or radio coverage, it is necessary to have a multi hop network and protocols that support multi hop function. In this case there may be different type of nodes that is mobile or static, in addition it is expected that node may have different computational capability and power. An adversary may have more computational capability and power, then nodes that are legitimate users of the network.

The network has to be protected from un-legitimate users, which may reduce the networks ability to support the mission, to support civilian in an area of conflict, ideal organisation as red-cross etc. If the network is not protected from un-legitimate user, unit or organisation that is part of the conflict may be able to misuse network resource. In addition, the ideal organisation may be accused to take part in this conflict. According to this, there is at least two reasons to assure that only legitimate user have access to the network, 1) To assure the networks ability to support the mission [39] and 2) protect the reputation to ideal organisation.

Today there are many different wireless networks such as WLAN, TETRA, GSM etc but they have a static backbone, which consist of base station, switches and control centre. The most used wireless system has radio coverage between 30-200 meters, and a channel capacity greater than 2Mbit/s [39]. It is assumed that nodes within MANET have some of same property as the standard IEEE 802.1x, but does not have a central control. Within this radio coverage it is possible for people and different kind of vehicle to operate as, on the ground (cars, motor bike, cycles), sea (boat) and in the air (helicopter etc.).

3.2 Functional

Every node that is used in MANET must have functionality to rout, transmit, receive, authenticated message and authenticate node that is joining the network. There have to be a routing protocol that supports to find the shortest path between nodes. Routing protocol is not discussed in this report.

3.3 Physical restriction

As mention earlier, node within a MANET may be small, have restricted power and computation capability, where the most of the power have to be used to receive or transmit message. This also restricts how long time a node may be used. To assure that the lifetime is not reduces by a security attack, it has to be protected. But security protocol has to be efficient comparing to power consumption, since this will reduce the life time of the node. To assure this, an authentication protocol that shall be used in a MANET should satisfy following condition [17]:

- Few computational steps
- Balanced computational steps
- Cheap computational step
- Few messages flow
- Small messages
- Small program memory
- Small data memory requirement
- Restricted consequences of data disclosure

In this master thesis it is assumed that nodes have a life time up to 5 years, restricted by the battery and rapidly evolution on technology.

3.4 Security issues

3.4.1 Authentication

Authentication is fundamental in information security and assurance, and is the binding of an identity to a subject. Authentication may be based on [25]:

- something known (as a password, shared secret, secret, the private key corresponding to a public key etc.)
- something possessed (this is typical a physical asset as a badge card, id-card, password calculator etc.)
- something inherent (handwrite, fingerprint, etc.).

An authentication protocol proves the nodes identity in a given instance of time. To maintenance the identity authentication additional techniques must be included. If nodes is authenticated at the start of a session, they have to ensure that they maintenance the authentication during the session, so that an adversary has not interfered the session. An approach [25] to prevent this to happen include:

- perform re-authentication or for each discrete resource request (eg each message that have to be exchanged
- tying the identification to an ongoing integrity service, that each message can be tied together with session authentication.

3.4.2 Threat

In general, an adversary can be divided into two groups 1) gain access to the network and use its resource and 2) damage or harm the network.

An attack may come from anywhere and from all direction. Attacks may includes passive eavesdropping over the wireless channel, active attack e.g. as denial of service attack by malicious nodes and attack from compromised nodes or stolen devices. A denial of service (DoS) attack can be launched at any layer in a mobile ad hoc network. On the physical and media access control, an adversary could employ a jamming to interfere with communication on physical layer. Adversary may [7] on the network layer disrupt the routing protocol and disconnect the network, which may be done by forging and malicious alternation of packet, or [31] by impersonate the sender and inject broadcast packets (because malicious packet injection is easy in many broadcast networks). The wireless channel [23] is more erroneous and loosely than wired network. That's mean it is difficult to distinguish between that a malicious node has dropped a packet, or it is caused by packet loss in the erroneous communication channel. An adversary [42] may inject a huge number of spurious packets into the MANET with the goal of depleting the resource of the nodes that relay the packets, as a resource consuming attack. The attacker may impersonate a legitimate node by insert the node's id in the source id field of the packets. To achieve that goal, an attacker may eavesdrop on other nodes, reorder or drop packets, replay packets, or modify overheard packets and re-inject them into network. Another serious attack is the tunnelling attack also called Wormhole attack [19, 18], where an adversary may record packet at one location and tunnel the packet (or selective packet from e.g. protocol execution or messages) to another location and replay it there. In this way two nodes may appears as they are neighbour, without to be within radio coverage of each other.
All these attacks can be summarised according to [25]:

- Impersonation: a deception whereby one entity purports to be another.
- Replay attack: an impersonation or other deception involving use of information from a single previous protocol execution, on the same or a different verifier. For stored files, the analogue of a replay attack is a restore attack, whereby a file is replaced by an earlier version.
- Interleaving attack: an impersonation or other deception involving selective combination of information from one or more previous or simultaneously ongoing protocol executions (parallel sessions), including possible origination of one or more protocol executions by an adversary itself.
- Reflection attack: an interleaving attack involving sending information from an ongoing protocol execution back to the originator of such information.
- Forced delay attack: a forced delay occurs when an adversary intercepts a message (typically containing a sequence number), and relays it at some later point in time. Note the delayed message is not a replay.
- Chosen text attack: an attack on a challenge-response protocol wherein an adversary strategically chooses challenges in an attempt to extract information about the claimant's long-term key. Chosen-text attacks are sometimes referred to as using the

claimant as an oracle, i.e., to obtain information not computable from knowledge of a claimant's public key alone. The attack may involve chosen-plaintext if the claimant is required to sign, encrypt, or MAC the challenge, or chosen-cipher text if the requirement is to decrypt a challenge.

Protection

There are three approaches to deal with security attack: prevention, detection and reaction. One way to protect [31] is to ensure that the broadcast packet really originate from the claimed source. To protect against resource consuming attack [42] is primary related to preventing from this kind of attack, secondary detection and reaction. Prevent an insider attack is a very difficult problem. Resource consuming attack compared to the channel jamming attack, which only affect a relative small area around the malicious node and may be addressed by techniques such as spread spectrum, channel surfing, or spatial retreat, the packet injection attack using broadcast message may be more favourable to an attacker due to its network-wide harm. To protect against jamming [38, 7] is an issue of the physical and data link layer, and is not part of this master thesis.

4 Background and related work

4.1 General

There is ongoing research [17, 18] to design a secure MANET. These research ranges from authentication protocol, organisation model, trust model to routing protocols. Where traditional cryptographic primitives as described in section 4.7 are used as tools to secure MANET.
Authentication protocol is used between nodes to prove theirs identity of each other. This protocol is used in different phases and layer e.g. in routing protocol. Public key cryptography is frequently used in MANET together with hash or MAC function. In sensor network, symmetric key cryptography is often used.

Organisation of authentication in MANET is mainly divided in two groups 1) a trusted authority or 2) in a self organisation manner. The case with a trusted authority is appropriate to be used in the scenario of the rescue operation, as there is always an organisation that has the responsibility of the operation. In futher section different earlyer proposed authentication protocol is discused.

4.1.1 Taxonomy to classify authentication protocol

In [1]a taxonomy is proposed, which may be used to analyse authentication protocol. The classification is based on authentication function, type of credentials and establishment of credentials.

Classification based on authentication function

These classification in figure 1 is a classification according to the network organisation, where the network is either homogeneous or heterogeneous network.

In a homogeneous network, nodes have the same role. Node may collaborate to establish authentication (in this case they depend on each other), or they does not collaborate (in this case they are autonomous).

In a heterogeneous, network nodes has different role. There may be a central node that provides service to establish authentication, this service can be distributed to one or more node within the network. In a cluster based protocol, each node belongs to a group or cluster, where there is established trust between nodes within a cluster.

Classification based on type of credential

Authentication protocol may be classified based on what kind of information nodes proposed to establish authentication. According to figure 2 authentication protocol may be based on identity or context.

An identity based authentication, is based on the unique identity of a node to prove its identity with high assurance. This may be based on encryption or non-encryption function.

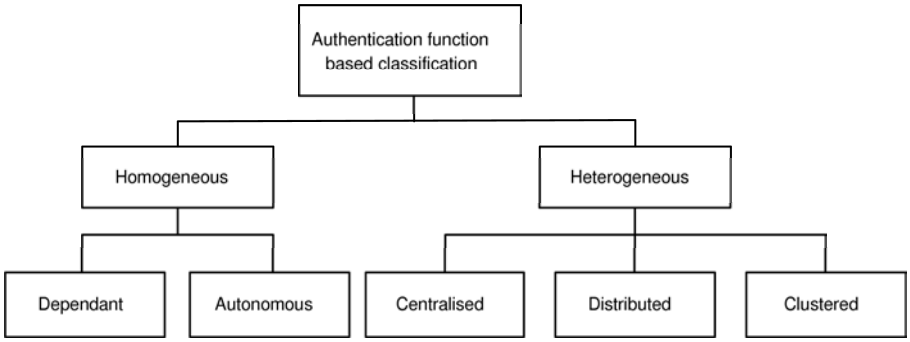

Figure 1: Classification based on Authentication function

A context based is more related to pattern, behavioural, or physical characteristic of a node. One example is authentication based on location, since a node may not occupy two or more different location in the same time, the location of the node may be used to identify the node. To decide the location based on signal property [35], two or more nodes have to collaborate to measure and compute the position (based on e.g. timing and signal strength). Collaborating node must be trusted, and they have to know its own location relatively to other nodes.

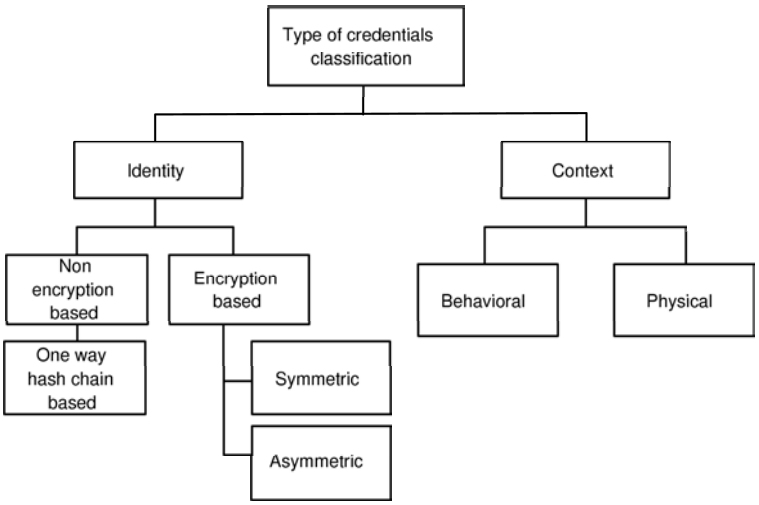

Figure 2: Classification based on type of credential

Classification based on establishment of credentials

The classification of protocol may also be based on how and when the credential is established; where the credential is used in the authentication process. The credential may be established before deployment, derived from credential that was pre deployed or after deployment of nodes, as in figure 3.

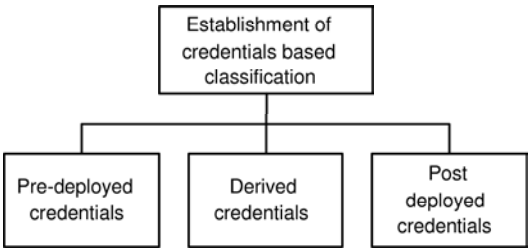

Figure 3: Classification based on establishment of credentials

4.1.2 Protocol phases

It is usually to divide authentication protocol in phases as Pre-distribution of credential, Boot-strap, Pre-authentication, Credential establishment, Authentication and Monitoring behaviour as in figure 4.

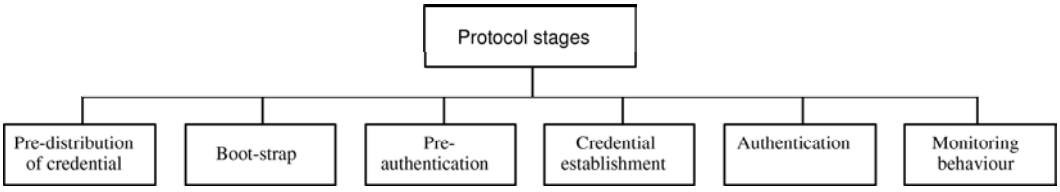

Figure 4: Protocol phases

Pre-distribution of credential

If the network based on full self organisation, there must be some spre-shared secret to authenticate nodes, which is distributed from a trusted third party (TTP). This may be a shared key based on threshold scheme, a secret group key, certificate, unique identity and certified public/private key.

Boot-strap

The phase that includes Pre-authentication, Credential establishment and Authentication are reguraly named as the Boot-strap phase, which is used the first time nodes prove its identity. In some protocol this phase, is also repeated when a node is re-joining, changing credential or moving to a new location.

Pre-authentication

In this phase a node is showing its credential to show who it is.

Credential establishment

The new node establishes new credential that the other node can verify and decide whether the new node is genuine.

Authentication

In this phase, a node is proving its claimed identity, if it correct and there is no dupe about it has proved its identity, the nodes is successfully authenticated.

Monitoring behaviour

To maintenance the trust, a node monitors each others behaviour to detect if there is some compromised node.

4.2 Earlier proposed authentication protocol

In this section, some resent proposed authentication protocols are presented, these protocols are grouped according to [1] as mention in section 4.1.1.

4.2.1 Heterogeneous protocol

These protocols use two different approaches. The first one is based on threshold scheme with a shared secret, where the secret can be shared to some central nodes or distributed to all nodes. The second is based on cluster organisation of node, where each node belongs to a group of nodes (named the cluster).

Threshold scheme

Threshold scheme, is based on [32] a secret, which is shared between n node. If k ($k < n$) or more nodes are collaborating, then it is easy to compute the secret. But if less then k nodes are collaborating it is hard to compute the secret. This is called the (k, n) threshold scheme. This scheme is frequently used together with the Lagrange interpolation [25]. Both homogenous and heterogeneous protocol based on threshold scheme that are proposed and discussed in [41, 24, 8, 40].

Heterogeneous distributed protocol

A heterogeneous distributed protocol is proposed in [41], where the private key from a certificate authority (CA) is divided in n shares, to each n server as a fixed group. Each server stores the public key that belongs to every node, and other servers within the network. It is assumed that every node has access to the public key that is according to the shared private key, from the CA. If Alice receives a message or has to authenticat message from Bob, Alice needs an authenticated copy of Bob's certificate/public key. Then she send a request to $k + 1$ servers. These servers generate a partial signature on Bob certificate/public key, and send the certificate/public key to Charli (a combiner). Charli combine these partial signatures, and send the full signature to Alice. In this way, Alice has access to authenticated copy of Bob's certificate and public key

This protocol requires a lot of computation and message exchanges, to verify a certificate or a signature. In addition, it requires that the servers stores the public key to every node, and that Alice have access to at least $k + 1$ server. In a MANET that is used in a rescue operation, there may be hard to meet these requirements, especially to have access to a number of servers and be able to do this computation for the combiner and server.

Cluster based

A cluster based authentication protocol is purpose in [2], which has more hierarchic structure. The protocol is based on private/public key and certificate, and includes a metric that is a value between 0.0-1.0. This value is the trust level between nodes. There are two trust values the direct trust and the indirect trust. Nodes that belong to the same cluster have a direct trust (value) among each other. If Alice has to connect to Bob, and Bob belongs to another cluster. Alice asks for the group, and send a request to the other node in the same cluster as Bob's, this nodes is the introducer to Bob. Each introducer replay with a message that contains Bob's public key and the trust value between that

introducer node and Bob, this message is signed with introducer private key. Alice collect message and compute the indirect trust value for Bob, and decide that Bob is trustworthy or not. This protocol, requires lot of public key operation, since each introducer have to do private/public key operation and the initiator have to verify each responder.

The use of a shared secret based on threshold scheme or cluster based may not be appropriate to be used in the scenario of a rescue operation, since the network topology may change continuously. Another draw back of using shared secret is that it is hard or impossible to up-date (if the shared secret is compromised) the shared secret when network is deployed. In a cluster based protocol, there may be hard to maintenance the trust relationship between nodes, when node is free to move and there may not be any direct connection between cluster nodes for some time. This is also some of the same conclusion mention in [17].

4.2.2 Homogeneous protocol

Dependant protocol

Threshold scheme [32] is also used in homogeneous protocol [24, 8], where nodes are collaborates in the authentication process and to establish shared secret. In the proposed authentication protocol [24] part of the private key from a CA, is distributed among node that is part of the network. And every node knows CA's public key. If at least k nodes is collaborating, then they have the same function as a CA. In addition each node has its own private/public key. The public key is included in the certificate that belongs to the node, and the certificate is only valid if it is signed by CA's private key. This is possible if at least k nodes is collaborating in the signing process. The certificate has a life time, before it is expired a node have to ask for a new certificate. The node, request at least k node for a new certificate to be issued, and receive k partial signed certificate. Before the k node is generating the partial signing on the new certificate, they check their record on this node. If the node as is trust fully the partial signature on the new certificate is generated.

Another approach is used in [8], where the source send a message in a secure way to k, which generate a partial signature on messages. When the destination node receive the partial signed message it combine this messages, if there is something wrong with the signature or less then k messages is received. The message is not authenticated.
To use this scheme [17, 24], there must be a pre-distribution of certificate and a shared secret among these n node, before the first time node is communication. According to [17] at least k have to be collaborate, to be able to authenticate nodes or message, which may not be possible in some scenario. According to [8], there has been done some performance testing on threshold scheme with RSA to sign message. But it requires lots of computation power and is not recommended to be used in MANET. One option is to use elliptic curve cryptosystem together with a threshold scheme, but this is also assumed to require more computational power then may be useful for MANET. Since it require, that every message is signed. Another drawback to this approach is that, it requires that the number of nodes is known before the shared secret is distributed and node is deployed.

Autonomous protocol

full self-organisation approach

In [6] considers a full self-organisation approach. This approach includes a certificate graph G (as a directed graph that only contain valid certificate) that contains vertices and a set of edges, where vertices represent public key and edges represent certificate. Where the graph G represent the entire network, which only includes valid certificated and public key. Each node store a certificate graph G_u and G_uN. Where G_u include all valid certificate and G_uN includes the non-updated certificate. Where G_u is a sub graph of G, but G_uN is not. If there is an edge from vertex K_u to K_w, there will also be a certificate that is signed by $u's$ private key that binds K_w to an identity of w. Each node produce it own certificate and public/private key. Node u and w may exchange identity information through physical contact, or with use of an authenticated channel. Since node is moving, it will be in contact with other nodes, in this way node may exchange authenticated identity and public key. If a node u wants to verify the authenticity of the public key to another node v, it tries to find a direct path in $G_u \cup G_v$. If there is no direct path, u tries to find a path in $G_u \cup G_u^N$. If there is a path u updates expired certificate and perform authentication. If there is no path, the authentication fails.
A full self-organisation approach for a MANET is not appropriate to be used in a rescue operation, since there is no control on who is allowed access to the network. It also requires a number of private/public key operations, when a node generates a signature to authenticate nodes identity and public key. This operation has to be repeated, when a previous authenticated node has change private/public key, and still is trusted.

single authority

The approach that has been used frequently is a combination of private/public key and hash function. Protocol proposed in [3, 19, 23, 38, 31, 42] is based on use of public/private key and hash chain, and late key disclosure according to [21]. Protocol [3, 19, 38, 31, 42] require loosely time synchronisation between nodes. The hash chain is used to authenticate message, since signing every message with the private key require lot of computation power and is time consuming.

In [3, 19, 23, 38, 31, 42] private/public-key cryptography is used in the bootstrap of authentication between nodes. In this process both time synchronisation and distribution of n'th hash chain value is done. This approach is vulnerable to DoS attack, since it requires some computation power to verify a signature.

Protocol [19, 42] is based on Timed Efficient Stream Loss-tolerant Authentication (TESLA) protocol [31], but [42] use TESLA protocol to up-date key and re-joining node which has been authenticated before. According to [42] TESLA protocol is proved to be secure if node is loosely time synchronic. Protocol in [19] is designed to protect against wormhole attack. The protocol in [23] uses an index to keep track of hash chain key instead of loosely time synchronisation.

Password authentication with insecure channel Lesli Lamport has proposed a password authentications that can be used with insecure communication channel [21], which is based on one-way function e.g. hash function. In this case with Alice and Bob,

Alice choice a random secret $x_0 = h_0$. Where the hash and key chain is generated according to equation 4.1 and 4.2 as presented in figure 5 in section 4.7.3.

$$h_1 = h(h_0),\ h_2 = h(h_1) = h(h(h_0)) \ \Rightarrow\ h_n = h^n(h_0) \tag{4.1}$$

$$k_0 = h_n,\ k_1 = h_{n-1}, \ldots, k_j = h_{n-j}, \ldots, k_n = h_0 \tag{4.2}$$

To obtain authentication between Alice and Bob, this is possible, if k_0 is pre-distributed from Alice to Bob over an authenticated channel. Then Alice may authenticate with Bob later, when she disclosure the next hash chain key. This approach, has been extended in many message authenticated protocol with a late key disclosure, in this case k_0^A is pre-distributed over an authenticated channel as in algorithm 1.

Algorithm 1 Message authenticated protocol with a late key disclosure

1: $A \longrightarrow B:\ \ M, MAC(M, k_j^A)$
2: $A \longrightarrow B:\ \ A, k_j^A$

One of the security issue, is related to active attack, where an adversary may delay the message until the disclosure time of hash key k_j^A **.** Then the Adversary is able to manipulate message and re-compute the MAC, without Bob may detect that the message is changed.

This kind of protocol is very fast, since message is authenticated with use of hash chain and message authentication code (MAC), and it only use public/private key in the bootstrap phase.

4.3 Authentication protocol based on hash chain

This technique is the most promising approach; let us examine some proposed protocols in the literature.

These protocols are based on one-way hash function, where the hash function is used to generate hash chain keys and authenticate disclosed key. The hash chain and hash chain key is explained in section 4.7.3. When a key is disclosed it is public and all nodes that have received previous key, is able to authenticate last disclosed key. When the other nodes apply the hash function on the last disclosed key, the result should be the same as previous disclosed key. If not, the last disclosed key is not authenticated.

4.3.1 TESLA

The TESLA protocol [31], is based on one-way hash function and hash chain. Where hash chain value represents a hash chain key, which is named TESLA key. The TESLA key is disclosed within a defined time slot. Where all messages sent during this timeslot, are authenticated by the TESLA key related to this timeslot. To authenticate received message, messages have to be stored until the TESLA key is disclosed. When the TESLA key is disclosed, the receiver is able to authenticate all messages that are received during this TESLA period. To keep track of the TESLA period, protocol require that all node is loosely time synchronicity according to [31]. This is done by private and public key operation. That is also done to commit the hash chain key, such that the n'th hash value (the first key) is signed.

4.3.2 Lightweight network access control protocol for ad hoc network

Lightweight network access control protocol for ad hoc network (LHAP), is a lightweight hop-by-hop authentication protocol [42] as presented in algorithm 2. The protocol is based on TESLA protocol, but instead of one hash chain it has two hash chains. The first hash chain is used as a TESLA key, to maintenance trust among neighbour nodes. The second hash chain is used to authenticate traffic among neighbour nodes. The TESLA key is used, in the key up-date message, to authenticate last used traffic key. The key up-date message is periodical broadcasted to neighbour nodes, and is authenticated by the next TESLA key.

Before a node is joining the network, the node has to generate a signature to the hash chain, to committing the hash-chain, which is done by a private and public key operation. When a node has to change TESLA or traffic hash chain, the same operation has to be repeated. This kind of operation is vulnerable to DoS attack, it is also vulnerable to replay attack according to [23].

LHAP protocol scheme is described as follows:

Algorithm 2 Lightweight hop-by-hop authentication protocol (LHAP)

1: $A \longrightarrow B: \quad Cert_A, Sign_A(A|K_A^T(0)|K_A^F(0)|T_A^T(0)|T_A^F(0))$
2: $B \longrightarrow A: \quad Cert_B, Sign_B(B|K_B^T(0)|K_B^F(0)|T_B^T(0)|T_B^F(0)), K_B^T(i-1), MAC(K_B^T(i)|K_B^F(j))$
3: $A \longrightarrow B: \quad K_A^T(i-1), K_A^F(J), MAC(K_B^T(i)|K_B^F(j))$

Before a node joins a MANET, it first computes two key chains: a traffic and TESLA key chain. Then it signs the key chains and broadcast them to its neighbour nodes, this is message nr 1 in algorithm 2. Message nr 2 is the responds to message nr 1. In message nr 2, node B does not compute a new digital signature. But include signature that was generated when node B joined the network, or when at least one of the key chains (traffic or TESLA key chain) has been changed. Node B include the latest used traffic and TESLA key, in message nr 2, which is authenticated by the key up-date message. Since node B is re-using its certificate and signature on the hash chain, it reduces the computation for the authentication. But this may be more vulnerable for replay attack.

Message nr 3 is the key update message, which is sent to the neighbour node to maintenance authentications and authentication of disclosed key. It is also used during re-authentication (re-joining), if nodes has been authenticated before. But if at least one of these key chains has been changed, the node has to sign this by compute a new digital signature. This may be done before new authentication take place.

4.3.3 Lightweight Authentication Protocol for Mobile Ad Hoc Network

The protocol in [23] presented in algorithm 3, which is a variant of LHAP protocol [42] and TESLA [31]. The protocol includes one hash-chain, which is used in the key-update message and message authentication. The protocol does not require clock synchronisation, but instead use an index related to the key chain. To commit the key chain among neighbour nodes a private/public key operation is performed, this kind of operation is vulnerable to DoS Attack. Since it only use one key chain, it requires a higher number of private/public key operation or larger key chains compare to LHAP [42].

Algorithm 3 Lightweight Authentication Protocol for Mobile Ad Hoc Network

1: $A \longrightarrow B:$ $Cert_A, Sign_A(A|h_n^A|H_A)$
2: $B \longrightarrow A:$ $Cert_B, Sign_B(B|h_n^B|H_B)$
3: $A \longrightarrow B:$ $M, MAC(M, h_{n-j}^A), r, index$ where $(index = j)$
4: $A \longrightarrow B:$ $A, h_{n-j}^A, index$ where $(index = j)$

4.3.4 Security consideration

Protocols in section 4.3.1, 4.3.2 and 4.3.3 use a public/private key operation to establish the hash chain key, which is used to authenticated future protocol execution. This operation requires some computational power and is vulnerable to DoS attack. In addition these protocol does not protect against wormhole or an insider attack (as a compromised node), since the authentication of packet is done hop by hop.

4.4 Routing

According to [9], there is an IETF working group for MANET. The purpose of the working group is to standardize IP routing protocol functionality suitable for wireless routing application, for both static and dynamic topology. In [11], the routing protocol performance and evaluation consideration is addressed with respect to traditional packet network, and what effect it has on design and network control protocol.

Up to now there has been established three RFC within this area as some experimental routing protocols, which is Ad hoc On-Demand Distance Vector (AODV), Optimized Link State and Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) routing protocol.

Ad hoc On-Demand Distance Vector (AODV) routing

AODV [12]is a routing protocol where each node (router) maintenances a routing table, which list all possible destinations within the network. Each entry in the routing table contains the address (identity) to the destination and the shortest known distance and its neighbour that is the first hop on the shortest path. If a new destination is needed, the node sends a Route Request (RREQ) through the network. If there is a path to that destination, the requesting node receives a Route Replies and up-date its route table. If an active link is broken, a Route Error message is sent to notify other nodes.

Optimized Link State Routing Protocol

This protocol have another approach [13],where each node selects a set of neighbour node as a multipoint relay (MPR). Only this MPR is responsible to forwarding control packet into the network, in form as diffusion. To provide an efficient flooding of control information and reduction of transmission that is required.

Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)

This protocol [14] is based on that each node is building a tree containing shortest path to reachable nodes. The protocol combine periodical and differential update to each neighbour, to minimise the overhead only part of the tree is distributed to neighbour node.

The AODV [12] is frequently used in simulation as described in [42, 23]. There are also some other internet draft [9] that's cover routing protocols, neighbour discovery, multicast and message format. The routing protocol will not be discussed in details here

as it is not the primary of the interest of this thesis.

4.5 The trust model

If two nodes have succeeded on authentication of each other, then there is established a trust relation ship between these nodes. This mean if Node A and B has done the authentication process they trust each other. That is also true if node B and C have authenticated. But this does not mean that node A and C will trust each other. If node A and C have to trust each other, they have to carry out the authentications process.

Further, it is assumed that every legitimate node has a certificate with a unique identity and public/private key pair that is distributed and signed by an off-line TTP/CA. Every legitimate nodes trust this certificate. The private keys are stored in a secure tamper proof area in the node, and are only known by its owner.

4.6 Clock synchronisation

One of the most important decisions in a network design especially in MANET, is whether node should have a synchronised clock with small deviation. In resent year, there are many design and developments in small Global Position system (GPS) receiver, which is also implemented on a single chip [36]. These GPS receiver is used in small mobile devices as PDA, cellular phone, etc. In the future, it is expected there will be more regularly to include GPS receiver in small mobile devices then today, and the price and power requirement is supposed to fall to level where it is cost efficient to be used in MANET. According to [30], it is possible in GPS to have a clock accuracy that is better than 340 nanoseconds (95 percent), for civilian use. But clock accuracy may be restricted by the GPS receiver and the number of GPS satellite.

Attack on the GPS signal may occur especially in a hostile environment as a battle space, but it is assumed that it only will harm a smaller area not the entire network.

4.7 Cryptography

Cryptographic algorithm is frequently used within authentication protocol and is fundamental to achieve a certain level of trust. In general, crypto algorithm has to be fast and at least hard to break (or impossible) in the time period where it is used to protect credentials. There exists a number of cryptographic algorithms, which may be divided in following classes with different properties:

1. Symmetric
2. Asymmetric
3. Non encryption based (e.g. hash function, MAC, HMAC etc.)

4.7.1 Symmetric crypto algorithm

Symmetric crypto algorithm is a kind of algorithm that uses the same secret key to encrypt and decrypt messages. In this case, the source and destination must share the same secret key. In a MANET, it is not possible to include a central key distribution centre (KDC). If only symmetric crypto algorithm is used in MANET there exist two choices:

1. Every node shares the same secret key, as a group key, that only belongs to node that is legitimate user.
2. There is a point-to-point secret key between two nodes.

The first choice requires that only one secret key is stored in each node, but if one node is compromised, the entire network is compromised. Another drawback is, if it is possible there will be time consuming to do a key change when node is deployed. The second choice requires that each node store $\frac{n(n-1)}{2}$ secret key. If it is assumed that every node may communicate with each other, since a MANET do not allow the use of any central key distribution centre. Similarly as in the first choice, if one node is compromised the entire network is compromised. In both choices, the secret key must be distributed over a confidential and authenticated channel, before execution of the authentication protocol and deployment of nodes.

This is the reasons that it is not recommended to only use symmetric cryptographic algorithms in MANET, normally it is used together with asymmetric cryptographic algorithms.

An example on one mutual authentication protocol that uses symmetric encryption algorithm [25] is described in algorithm 4.

Algorithm 4 Authentication protocol based on symmetric encryption

1: $A \longrightarrow B:\ r_A$
2: $B \longrightarrow A:\ E_K(r_B, r_A, A^*)$, where A^* is optional field
3: $A \longrightarrow B:\ E_K(r_A, r_B)$

In this protocol, node A and B share the same secret key K, and r_A and r_B is random number or a nonce. This is a fast authentication protocol and do not require much computational power, but as mention earlier it is not recommended to be used in MANET.

4.7.2 Asymmetric crypto algorithm

In this system, it includes a private and public key, where the public key may be known by every one, but the private key is stored in a safe place at the owner or as for MANET within the node.

One example on mutual authentication protocol, which uses asymmetric encryption algorithm is the X.509 based mutual authentication protocol [25] in Algorithm 5.

Algorithm 5 Mutual authentication protocol based on asymmetric encryption

1: $A \longrightarrow B:\ r_A$
2: $B \longrightarrow A:\ cert_A, r_B, A, Sign_B(r_B, r_A, A)$
3: $B \longrightarrow A:\ cert_B, B, Sign_A(r_A, r_B, B)$

In this protocol, r_A and r_B are random numbers. The certificate includes the identity, public key etc that belong to the node, the certificate is signed by a trusted third party (TTP). The signature is taken over hash value of the random numbers and identity. To generate the signature, A and B use its own private key, to show that node A and B know its own private key. We simplify the notations in algorithm 6.

Algorithm 6 Mutual authentication protocol based on asymmetric encryption

1: $A \longrightarrow B:\ r_A$
2: $B \longrightarrow A:\ cert_A, r_B, A, Sign_B(m_B)$
3: $B \longrightarrow A:\ cert_B, B, Sign_A(m_A)$
4: Where $m_A = (r_A, r_B, B)$ and $m_B = (r_B, r_A, A)$.

Asymmetric crypto algorithm is divided in two group represented by these two difficulty problems:

1. it's hard to factoring large integer
2. it's hard to solve the discrete logarithm problem

RSA based crypto system

These cryptosystem is based on the difficulty to factoring large integer. A typical signature protocol from NIST [29, 27, 25] is the RSA signature protocol as described in algorithm 7:

Algorithm 7 RSA based crypto system

Key generation for RSA:
Generate two large distinct random numbers primes p and q, each rougly the same size. Compute $n = pq$ and $\phi = (p-1)(q-1)$. Select a random integer e, $1 < e < \phi$, such that $\gcd(e, \phi) = 1$.Use the extended Euclidean algorithm to compute the unique integer d, $1 < d < \phi$, such that $ed \equiv 1 \mod \phi$. A's public key is (n, e); A's private key is d. Entity A signs message $m \in M$. Any entity B can verify A's signature and recover the message m from the signature.

Signature generation. A does following:
1: Compute $\tilde{m} = R(m)$, where $R(\cdots)$ may be a hash function
2: Compute $s = \tilde{m}^d \mod n$
3: A's signature for m is s

Verification: B does following:
4: Obtain A's authentic public key (n,e)
5: Compute $\tilde{m} = s^e \mod n$
6: Verify that $\tilde{m} = M_R \mod n$, if not, reject the signature.
7: Recover that $m = R^{-1}(\tilde{m})$, where $R(\cdots)$ may be a hash function

According to the table 25 in the appendix, the complexity of doing the signature and verification for an RSA base protocol is:

Signature: Compute $s = m'^d \mod n$ has the complexity $O(\log_2(d) * \log_2^2(n)) \approx O(k * l^2)$, where k is number of bit in the integer d and l is number of bit in n. As mention above d should be chosen to be large, so the complexity is more likely to be $O(l^3)$.

Verification: Compute $m' = s^e \mod n$ is $O(\log_2(e) * \log_2^2(n)) = O(j * l^2)$, where j is number of bit in the integer e and l is number of bit in n.

In the case of signature and verification, the private key d should be chosen to be large according to the size of ϕ, but the public key e may be chosen to be much smaller than ϕ. With this assumption the, verification will be much faster then signing a message. But it's important to notice that it is opposite for encryption of a message when using RSA.

ElGamal based crypto system

This crypto system is based on the difficulty to solve discrete logarithm. The ElGamal signature protocol [25, 27] is described in algorithm 22 and have the computation complexity according to table 28 and 29 in the appendix. Another protocol that is based on ElGamal, is the Digital Signature Algorithm (DSA) that is recommended by NIST [27] and presented in algorithm 8 according to [25].

Digital Signature Algorithm (DSA) or Digital Signature Standard (DSS)

Algorithm 8 Digital Signature Algorithm (DSA)

Key generation for DSA:
Each node creates a public key and corresponding private key. Each node A does the following. Select a prime number q such that $2^{159} < q < 2^{160}$. Choose t so that $0 < t \leq 8$, and select a prime number p $2^{511+64t} < p < 2^{512+64t}$ with the property that q divides $(p-1)$. Select an element $g \in Z_p^*$ and compute $\alpha = g^{(p-1)/q} \bmod p$ if $\alpha = 1$ go to the previous step and repeat until not equal. Select a random integer a such that $1 \leq a \leq q-1$. Compute $y = \alpha^a \bmod p$. A's public key is (p, q, α, y), A's private key is a.

DSA signature generation and verification:
Node A signs a binary message m of arbitrary length. Any node B can verify this signature by using A's public key.

Signature generation, node A does following:
1: Select a random integer k, $0 < k < q$
2: Compute $r = (\alpha^k \bmod p) \bmod q$
3: Compute $s = k^{-1}(h(m) + ar) \bmod q$
4: A's signature for m is the pair (r, s).

Verification: to verify A's signature (r, s) **on** m**, B does following:**
5: Obtain A's authentic public key (p, q, α, y)
6: Verify that $0 < r < q$ and $0 < s < q$, if not, then reject the signature.
7: Compute $w = s^{-1} \bmod q$ and h(m).
8: Compute $u_1 = wh(m) \bmod q$ and $u_2 = rw \bmod q$
9: Compute $v = (\alpha^{u_1} y^{u_2} \bmod p) \bmod q$
10: Accept the signature if and only if $v = r$.

The table 26 and 27 in the appendix is the complexity analysis for each step in the DSA protocol, the complexity of:

Signature is of order $O(\log_2(q) * \log_2^2(p))$

The verification is of order $O(2 * \log_2(q) * \log_2^2(p) + \log_2^2(p)) = O(\log_2^2(p)(2 * \log_2(q) + 1)) \propto O(2 * \log_2(q) * \log_2^2(p))$

From the above, we know that the signature generation is approximate twice as fast as the verification. That is the same as discovered in [5] presented in table 4, but in [37] present in table 1 and 2 the timing of signature generation and verification is nearly the same. These tell us that the difference in computation complexity between signature generation and verification, depend on the implementation of the algorithm.

Elliptic curve system

Elliptic curve system is based on the difficulty of solving elliptic curve (in some literature this is also considered as discrete logarithm problem). This crypto system is based on point and point computation on an elliptic curve. Since the secret key is multiplied with a point on the elliptic curve, these crypto system is proven to have the same security with about $1/10$ [27] of the key sizes compared with RSA, DSA and ElGamal. Explanation to Elliptic curve is presented in A.5 that is based on [34]

The Elliptic Curve Digital Signature Algorithm (ECDSA) protocol [29, 34, 4] is described in algorithm 9. Some test results from [5] are presented in table 4. In ECDSA to execute modulus operation is very fast because of the small key size, but to do the point computation (which can be done by multiplication and addition of point) is the time consuming part of the algorithm.

Algorithm 9 Elliptic Curve Digital Signature Algorithm (ECDSA)

Initialisation:
Let p be a prime or power of two, and let E be an elliptic curve defined over F_p. Let A be a point on E having prime order q, such that the Discrete Logarithm problem in A is infeasible. Let $P = \{0,1\}^*$, $A = Zq^* x Zq^*$, and define: $K = \{(p, q, E, A, m, B) : B = mA\}$. Where $0 \leq m \leq q-1$. The values p, q, E, A and B are the public key, and m is the private key. For $K(p, q, E, A, m, B)$, and for a (secret) random number k, $1 \leq k \leq q-1$, define:

Signature generation:
1: $sign_K(x, k) = (r, s)$
2: $kA = (u, v)$
3: $r = u \bmod q$
4: $s = k^{-1}(SHA-1(x) + mr) \bmod q$

To be computed:
5: $x_3 = \lambda^2 - x_1 - x_2 \bmod p$
6: $y_3 = \lambda(x_1 - x_3) - y_1 \bmod p$
7: $\lambda = (y_2 - y_1)(x_2 - x_1) - 1 \bmod p$, if $P \neq Q$
8: $\lambda = (3x_1^2 + a)(2y_1) - 1 \bmod p$, if $P = Q$

Verification of signature:
9: $w = s^{-1} \bmod q$
10: $i = wSHA-1(x) \bmod q$
11: $j = wr \bmod q$
12: $(u, v) = iA + jB$

To be computed:
13: $x_3 = \lambda^2 - x_1 - x_2 \bmod p$
14: $y_3 = \lambda(x_1 - x_3) - y_1 \bmod p$
15: $\lambda = (y_2 - y_1)(x_2 - x_1) - 1 \bmod p$, if $P \neq Q$
16: $\lambda = (3x_1^2 + a)(2y_1) - 1 \bmod p$, if $P = Q$
17: $ver_K(x, (r, s)) = true$ if $u \bmod q = r$

4.7.3 MAC, Hash functions and hash chain

Hash functions

A hash function is a one way function that is hard (or impossible) to invert and is used to creating a small digital "fingerprint" from any data. A hash function has to satisfy following properties [33]:

1. The hash function have to be applicable to a block of data of any size
2. The hash function should produce a fixed-length output
3. The function $h(x)$ is relative easy to compute for an given x, making both hardware and software implementation practical

4. For any given value y, it is computationally infeasible to find x such that $h(x) = y$ (the one-way property).
5. For any given block x, it is computationally infeasible to find $y \neq x$ such that $h(y) = h(x)$ (the weak collision resistance)
6. It is computationally infeasible to find any pair (x, y) such $h(x) = h(y)$ (the strong collision resistance)

Hash chain

With the hash function it is possible to make a hash chain according to figure 5.

$$\begin{aligned} &h_1 = h(h_0), \text{ where } h_0 \text{ is the starting value} \\ &h_2 = h(h_1) \\ &\vdots \\ &h_{j+1} = h(h_j) \\ &\vdots \\ &h_N = h(h_{N-1}) = h^N(h_0) \end{aligned} \tag{4.3}$$

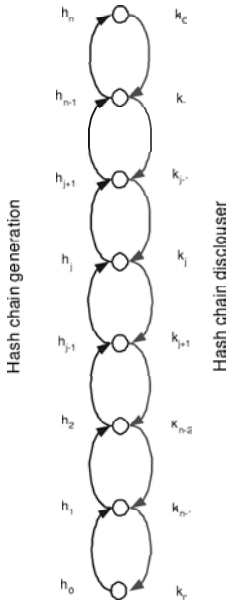

Figure 5: Hash chain and keys

The hash chain may be used in an authentication protocol according to algorithm 15 in appendix A.1.2. When a hash chain used as a key chain, the last hash value h_N is the first key k ($k_0 = h_N$) and the last key is the first value that was use to compute the first hash value ($k_N = h_0$). The next hash chain key k_{n+1} can be authenticated by the current hash chain key k_n, but the key (from the hash chain) may only be used one time to authenticate a message. Since when the key is disclosed, it is public (or known by more then one node). The hash function has to satisfy property mention earlier, and it is computationally infeasible to compute the next hash chain key k_{n+1} if the current k_n is known. Therefore, key that is not disclosed has to be protected as a secret key until it is disclosed. h_0 has to be protected since it is used to generator the entire hash chain.

Message Authentication Code (MAC)

A message authentication code is used to authenticate message that is transmitted as presented in figure 6. The MAC function has two input parameters, the message m and a secret key k. The output from the MAC function has a defined length (in number of bit) independent regarding the size of input data, and is a fingerprint of the message and key. A MAC function has to satisfy the same property, as for hash function. Only the user that has the correct key may generate the valid MAC, in this way it is used to assure that the message is authenticated. There exist several functions such as HMAC based on SHA-1 or MD-5, CBC-MAC based on AES and so on. These functions will not be discussed here in detail, but it is important to note that these functions are fast.

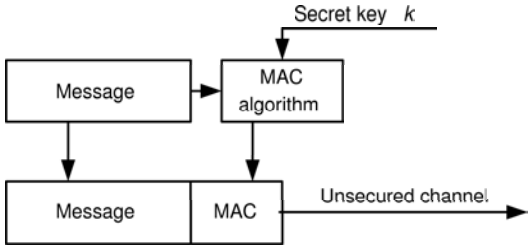

Figure 6: Message Authentication Code (MAC)

4.7.4 Conclusion on cryptography

In Table 36 and 37 in the appendix the recommendation from NIST is present [27] of key size to RSA, DSA, SHA-1 and ECDSA. In this thesis, a key size that is according to 80 bit security is assumed to be proper choice to be used in MANET, because of the lifetime of equipment is assumed to be less than 10 years and there should not be any reasons to protect authentication credential after end of life time of a equipment. In [37, 5] some results from test on several crypto algorithm are present.

According to [37], a benchmarks of different crypto algorithm on Pentium 4 2.1 GHz processor under Windows XP SP 1 and AMD Opteron 1.6 GHz processor under Linux 2.4.21, are presented in table 1 and 2.

Function	RSA(1024)	DSA (1024/160)	SHA-1 (160-bit)
Signature	4.75ms	2.18/1.13ms(with pre-computation)	
Verification	0.18ms	2.49/1.79ms(with pre-computation)	
Process 160 bit			0,281µs

Table 1: Benchmark of RSA, DSA and SHA-1 (Pentium 4 2.1 GHz processor)

Function	RSA(1024)	DSA (1024/160)	SHA-1 (160-bit)
Signature	2,07ms	0,8/0,48ms(with pre-computation)	
Verification	0,07ms	0,91/0,78ms(with pre-computation)	
Process 160 bit			0,190µs

Table 2: Benchmark of RSA, DSA and SHA-1 (AMD Opteron 1.6 GHz processor)

Function	MS XP (Megabyte/sec)	Linux 2.4.21 (Megabyte/sec)
HMAC/MD5	216,67	152,38
CBC-MAC/AES (Rinjndael)	57,57	47,94
MD5	216,67	152,38
SHA-1	67,98	100,20
AES-128 (Rijndael 128-bit key)	61,01	49,61

Table 3: Benchmark of HMAC/MD5, CBC-MAC/AES, SHA-1 and AES-128

There have also be done other test [5] of RSA, DSA and ECDSA on different micro-processor platform. Some result from [5], are present in table 4 for RSA with 1024 bit key, DSA with $q = 160$ and $p = 1924$, and ECDSA based on Koblitz curves over $F_{2^{163}}$. These tests are done on three different platforms, Pager with 32 bit data buss and 10MHz clock, Pilot with 32 bit data buss and 16 MHz clock, and Pentium II with 32 bit data buss and 400 MHz clock.

Encryption algorithm	*Pager*	*Pilot*	*PII*
DSA key gen	-	-	54.674,00
DSA signing	9.529,00	25.525,00	24,28
DSA verifying	18.566,00	52.286,00	47,23
RSA key gen	580.405,00	1.705.442,00	2.740,87
RSA signing	15.889,00	36.130,00	66,56
RSA verifying($e = 3$)	301,00	729,00	1,23
RSA verifying($e = 17$)	445,00	1.058,00	1,76
RSA verifying($e = 2^{18} + 1$)	1.008,00	2.374,00	3,86
ECDSA key gen	751	1.334,00	1,47
ECDSA signing	1.011,00	1.793,00	2,11
ECDSA verifying	1.826,00	3.263,00	4,09

Table 4: Test of RSA (1024), DSA (160/1024) and ECDSA (163), result is in ms

According to table 1, 2, 3 and 4, test that has been done to compare the speed of these protocol (SHA-1, RSA, DSA and ECDSA), we may conclude according to table 5. If we only consider the computation complexity; if the authentication protocol requires that the signature generation has to be fast then ECDSA is the best choice, if it is important that the verification of a signature has to be fast then RSA is the best choice. But hash function (e.g. SHA-1) and MAC (HMAC) function is even faster, if possible signature generation and verification should be replaced with hash or MAC function to save power.

Function	*RSA(1024)*	*DSA (160/1024)*	*ECDSA (160)*
Signature generation	Very slow	Slow	Fast
Verification	Fast	Very slow	Slow

Table 5: Comparing of RSA, DSA and ECDSA

5 First proposed authentication protocol and its analysis

5.1 General

The motivation is to establish a secure and efficient authentication protocol, which can be used in a MANET. In chapter 4, there have been discussed different crypto algorithms and how to do authentication. According to section 3.1 MANET used during a rescue operation, only legitimate user (node) may join the network. In this case there have to be some way to control who that is joining the network. In this scenario there is not possible to estimate how many nodes that will participate in the network. With this constraint, authentication protocol based on symmetric or pre-shared secret is not appropriate to be used. Since it requires that the organisation know how large the network will be or grow, and if one node is compromised the entire network may be compromised.

With the use of asymmetric crypto algorithm, there is possible to pre-distribute a unique credential to every node, which can be used to prove that a node is a legitimate user. Since asymmetric crypto algorithm is computation costly, it may not be used to maintenance authentication during a session or message exchange. Other algorithm that is more appropriate is MAC and hash algorithm, which is nearly or as efficient as symmetric algorithm according to table 3. Asymmetric crypto algorithm may also be used to distribute credential, which may be used to maintenance authentication during session and message exchange. In this way, the distributed credential can replace public/private key operation on each message that is exchanged. The distribution of credential may be done in two ways:

- encrypt the credential by the receiver nodes public key: this option requires that the initiator know the public key to the receiver node that is within radio coverage, and it have to do this operation to each node that shall be authenticated.
- the credential has a signature base on the private key to the node, and broadcast this credential to every node within radio coverage. If this message also includes initiators public key and its unique identity, which is signed by a trusted third party, all nodes within radio coverage is able to verify the credential and which is a legitimate node.

The first option require more message exchange during authenticate of its neighbour nodes, than the other option. The second option requires that the credential which shall be used have to be public, and there must be possible to maintenance the security even when the credential is public. With the use of a one-way hash chain, it is possible to satisfy this requirement. According to Lamport [21], if the $n'th$ hash chain value is distributed over an authenticated channel from a initiator to a responder. It is possible to use the $(n-1)$ hash value to authenticate message. Even when the $n'th$ hash chain value is done public, it is hard to find the $(n-1)$ hash chain value as discussed in section 4.7.3.

As we know the energy consumption E is [15]:

$$E = \int_0^t P(t)dt \tag{5.1}$$

If we expect that the power needed for an operation is constant during the time to execute the operation. It is easy to see that the energy needed to execute the operation, is proportional to the time it requires.

The energy consumption is dominated by the number and length of message that is transmitted. In addition the energy consumption is proportional with computation complexity to the algorithm (it is assumed when an operation is more complex to compute, it require more time, and consume more energy). There must also be assumed that to do a computation require less power than to transmit a message.

There also exists some other authentication protocols, such as Zero knowledge protocol in algorithm 18 and 20 in the appendix. But according to table 33 in the appendix, these protocol are also computational costly, and do not include an efficient way to maintenance authentication during session or message exchange.

5.2 Authentication protocol (hop-by hop authentication protocol)

Based on the discussion in chapter 3 and 4 we try to design a new authentication protocol, which is a combination of protocol in [31, 42, 23] as present in section 4.3. The proposed protocol is a hop-by-hop authentication protocol that includes asymmetric algorithm and two hash chain. In a hop-by-hop authentication protocol, messages are authenticated at every hop between source and destination node, where each neighbour (which is within radio coverage) is authenticated. The protocol is present in algorithm 10 and figure 7 below. This protocol has several security weaknesses as vulnerable to replay and DoS attack, this will be discussed in the security analyses.

The protocol in algorithm 10 requires that every node in the network have two hash-chains. A master hash chain K_A^M and a second hash chain K_A^T (the index A note that it belong to node A). The first master hash chain key and the valid time period to K_A^M is signed by node A's private key, which is according to the public key in the certificate. The idea is that the master hash chain shall replace public/private key operation, and be used next time the same node is doing the authentication process. It is also used to authenticate traffic hash chain key K_A^T and maintenance authentication. The hash chain K_A^T is used as a temporary traffic key to authenticate every message from node A to its immediately neighbour node.

5.3 Description of hop-by-hop authentication protocol

The protocol is divided into four phases represented as 1) Phase 1 pre-distribution of certificate and generation of hash chain K_A^M, 2) Phase 2 the bootstrap phase and change master hash chain, 3) Phase 3 maintenance authentication and message authenticate message and 4) Phase 4 Re-authentication and change traffic hash chain. It should also [1] include monitoring that have to be operational during phase 2-4 to detect fault and attack. Every message within these phases is broadcasted to the neighbourhood.

Algorithm 10 Hop-by-hop authentication nr 1

Phase 2 Boot-strap and change master hash chain:

1: $A \rightarrow B : Cert_A, Sign_A(A|K_A^M(0)|T_A^M(0)),$
$n_A, K_A^M(i_A - 1), K_A^T(j_A), MAC(n_A|K_A^M(i_A)|K_A^T(j_A))$

2: $B \rightarrow A : Cert_B, Sign_B(B|K_B^M(0)|T_B^M(0)),$
$n_B, K_B^M(i_B - 1), K_B^T(j_B), MAC(n_B|K_B^M(i_B)|K_B^T(j_B))$

3: $A \rightarrow B : A, K_A^M(i_A), K_A^T(j_A + n_B), MAC(K_A^M(i_A + 1)|K_A^T(j_A + n_B))$

4: $B \rightarrow A : B, K_B^M(i_B), K_B^T(j_B + n_A), MAC(K_B^M(i_B + 1)|K_B^T(j_B + n_A))$

Key up-date:

5: $A \rightarrow B : A, K_A^M(i_A - 1), K_A^T(j_A + n_B), MAC(A|K_A^M(i_A)|K_A^T(j_A + n_B))$

6: $B \rightarrow A : B, K_B^M(i_B - 1), K_B^T(j_B + n_A), MAC(B|K_B^M(i_B)|K_B^T(j_B + n_A))$

Phase 3: Phase 3: Maintenance authentication and message authentication:

7: $A \rightarrow B : A, m_A, MAC(A|m_A|K_A^T(j_A + n_A + 1))$

8: $A \rightarrow B : A, K_A^T(j_A + n_A + 1)$

Key up-date:

9: $A \rightarrow B : A, K_A^M(i_A), K_A^T(j_A + n_A + 1), MAC(K_A^M(i_A + 1)|K_A^T(j_A + n_A + 1))$

10:$B \rightarrow A : B, K_B^M(i_B), K_B^T(j_B + n_A), MAC(K_B^M(i_B + 1)|K_B^T(j_B + n_A))$

Phase 4: Re-authentication and change traffic hash chain:

11:$A \rightarrow B : A, K_A^M(i_A), K_A^T(0), MAC(A|K_A^M(i_A + 1)|K_A^T(0))$ (Chan. traf. hash chain)

12:$B \rightarrow A : B, K_B^T(j_B), MAC(B|K_B^M(i_B + 1)|K_B^T(j_B)$ (ACK)

Key up-date:

13:$A \rightarrow B : A, K_A^M(i_A + 1), K_A^T(0), MAC(A|K_A^M(i_A + 2)|K_A^T(0))$

14:$B \rightarrow A : B, K_B^M(i_B + 1), K_B^T(j_B), MAC(B|K_B^M(i_B + 2)|K_B^T(j_B)$

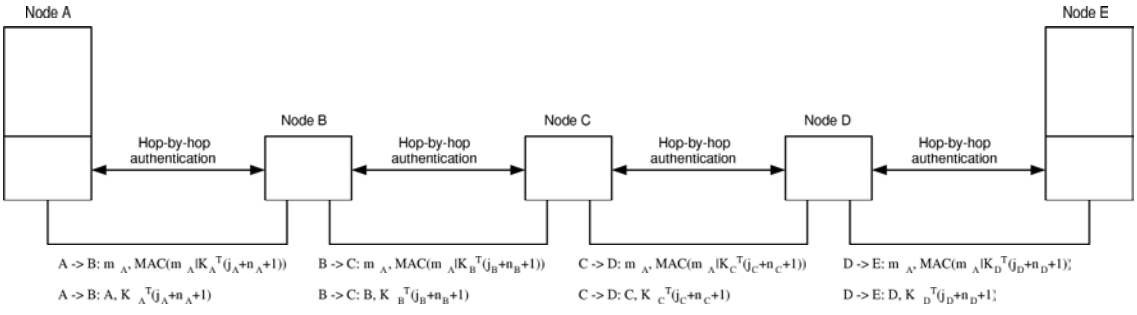

Figure 7: Hop-by-hop authentication protocol

5.3.1 Phase 1: Pre-distribution of certificate and keys

In this phase, a node receives, generates and store credential before the node is deployed and joining the network at the first time. The nodes get the certificate with a unique identity and public key from a TTP, the corresponding private key is stored in a safe place (tamper proof area) in the node. The node generate the master hash chain K_A^M, where the first mater hash chain key $K_A^M(0)$ and valid time period $T_A^M(0))$ is signed by node A's private key. The traffic hash chain (K_A^T), has also to be generated before the node is joining the network.

5.3.2 Phase 2: Bootstrap process and change master hash chain

In this phase, the node proves its identity and commits the master hash chain. All nodes have to go through this phase the first time it is proving its identity to another node, or when the node has to change the master hash chain.

Message 1

Node A broadcasts a message to its neighbour node. The message contains A's certificate, identity, the first master hash chain key $K_A^M(0)$ and the valid time period $T_A^M(0)$ to $K_A^M(0)$ (start time and end time). The id, first master hash chain key and the valid time period is signed by A's private key. The message also includes last used master key (current) K_A^M, K_A^T and an index n_A. The MAC is computed based on the next master hash chain key to K_A^M. The index n_A is used to be assure that node B have generated the traffic hash chain (K_B^T), to be assure about that node B have to disclose some part of this hash chain according to n_A.

Message 2

Message 2 is a response to message 1, and include: B's certificate, identity, the first master hash chain key $K_B^M(0)$ and the valid time period $T_B^M(0)$ to $K_B^M(0)$ (start time and end time). The id, first master hash chain key and the valid time period is signed by B's private key. The message also includes last used master key (current) K_B^M, K_B^T and an index n_B. The MAC is computed based on the next master hash chain key to K_A^M. The index n_B is used to be assure that node A have generated the traffic hash chain (K_B^T), to be assure about that node A have to disclose some part of this hash chain according to n_B. Before B decide to compute and send message 2, it verify that $K_A^M(i_A - 1)$ is according to the signed first master hash chain key $K_A^M(0)$, that it's valid according to $T_A^M(0)$ and the certificate is not expired.

If Node A received more than on responds (message 2), node A must decide how much of the traffic hash chain it should disclose, it should be the same or more than the lowest value that is expected from responding nodes. Node A have to verify the received master hash chain key $K_B^M(i_B - 1)$ according to the signed hash chain key $K_B^M(0)$, and that the certificate is not expired.

Message 3 and 4

In message 3 and 4, the node disclose some part of the hash-chain K_A^T, according to index n_A (for node A) or n_B (for node A) to prove that they really want to communicate with each other.

Key up-date

Node A and B sends a key up-date message immediately after message 3 and 4, which is according to message 5 and 6.

Verification of nodes

1. Node A and B verify that the key up-date message (message 5 and 6) is according to the value in message in message 1 and 2.
2. Node A and B verify the MAC value in message 3 and 4, is according to the key up-date message (message 5 and 6).
3. Node A and B verify that the hash chain in message 3 and 4, is according to value in message 1 and 2.
4. If all verification in 1-3 is correct, node A and B verify the signature and certificate in message 1 and 2.
5. If all item 1-4 is correct, then the node is authenticated.
6. The authentication between neighbour nodes is maintenance by every node is periodical sending a key up-date message.

5.3.3 Phase 3: Maintenance authentication and message authentication

The first part must be efficient to transporting data through the network, without time consuming computation to authentication the message at every node on the path between source and destination node. This is also the reasons the protocol is based on MAC and hash chain, since they are nearly or as efficient as symmetric algorithm according to table 3.

Message 7

Is used to exchange message and authenticated by a MAC, that includes the identity, message m_A and the next traffic key K_A^T as input parameters.

Message 8

After message 7 is transmitted, the next traffic key K_A^T is immediately disclosed in message 8.

Message 9 and 10

The key up-date (message 9 and 10) is periodical distributed within the neighbourhood, to maintenance authentication and authenticate traffic keys. To save energy node should only transmit messages when it has something to send or route. In this case, the last key up-date message is sent to its neighbour one key up-date period after the last message.

5.3.4 Phase 4: Re-authentication and change traffic hash chain

According to message 9 and 10 in phase 3, there are possible that nodes within the neighbourhood do not communicate continuous. Node may also re-join previous authenticate node, if the node or some nodes have left this part of the network for a time period. Then they have to re-authenticate. If a node after a period of time, have to send a message to its neighbourhood or to nodes that is already authenticated. They have to re-authenticate, to tell the neighbourhood what is the last used traffic key and the valid master key. This is also the case if a node has to change the traffic hash chain. The new traffic hash chain has to be committed and authenticated before the neighbourhood may trust this hash

chain. In this protocol, the new traffic hash chain is committed (signed by the master hash chain key) and authenticated by the master hash chain key.
In this way, private/public key operation is only need the first time nodes are authenticating each other, or when the master hash chain has been changed.

Message 11-14

If the node has to change its traffic hash chain, it has to send message 11 to its neighbourhood, and look for responds as in message 12. The change traffic hash chain has to be authenticated by the next key up-date message 13.

Key up-date

This message is also used when nodes are re-joining after a period of time. In this case nodes are already been authenticated in phase 2. But if the master hash chain has been changed they have to go through phase 2, to commit and authenticate the new master hash chain.

5.4 Security analyse of hop-by-hop protocol

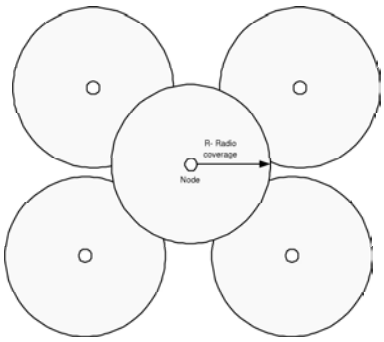

Figure 8: The neighbourhood

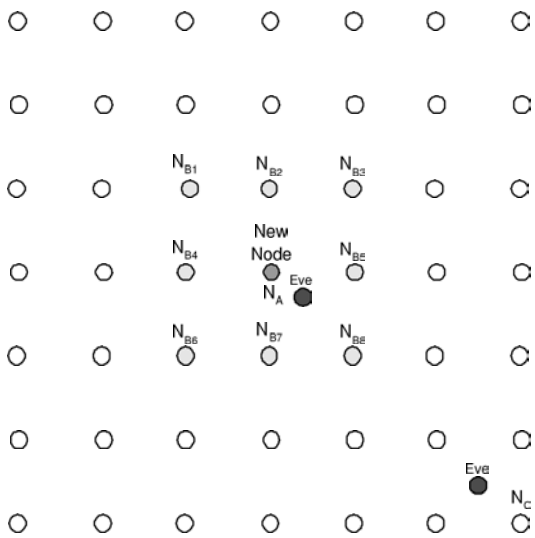

Figure 9: Replay attack at another location

In this protocol message is not encrypted they is public, and does not require a central

control. So there not much a passive adversary may gain from listening to the protocol execution. But an active adversary may interfere with the authentication protocol. In the security analyse, we assume an active adversary, witch may manipulate, delay and delete messages or part of a message. In the security analyses each phases is considered.
In the security analyses we use that initiator is named Alice (node A), responder is Bob (node B) and the adversary is Eve (node E). The adversary Eve may consist of one or more nodes, which may collaborate.

5.4.1 Phase 1: Pre-distribution of certificate and keys

It is assumed that the TTP is trusted by all nodes, and since the TTP is off-line, there will not be any attack that may be launched from the network. It's also assumed that TTP is placed in a safe area, and only authorised personal has access to the secret area of TTP. The private key to TTP is stored in a tamper proof area.

When the node received the credential from TTP, it generates and stores the master hash chain and signs the $n'th$ hash value $h_N = k_0$. This may also happened later. Before keys are disclosed, they are stored in a tamper proof area within the node. Also the private key is stored in a tamper proof area. It is assumed that an adversary does not have access or able to launch an attack during this phase, if this was the case it will compromise the trust.

5.4.2 Phase 2: Bootstrap process and change master hash chain

When a new node A is joining the network, it sends the join message that include certificate, root value to the master hash chain, the time period (start time and end time), the valid master hash chain. It also includes the last used key from the master hash chain and the traffic hash chain, and a challenge to the neighbour node. The challenge n_A informs the neighbour how much of its traffic hash chain that have to be disclosed in the response. Neighbour node is also sending the same kind of information back to node A, to tell node A that it may join node B.

When nodes receive the key up-date message, they can verify both the master hash chain key and the key that is used as challenge and evidence, it also ties the previous message together, so neighbour knows the last disclosed key from the master hash chain and traffic hash chain.

Delay attack

Since every neighbour node is within radio coverage of node A, there is not possible that Eve may delay an ongoing protocol execution between node A and its neighbour. Eve may try to replay earlier message to impersonation node A or some of its neighbour, but may only replay hash key that is already disclosed and known by the neighbourhood.

Replay attack

Eve may inject invalid packet into the network in some way that nodes is dropping all received packet or launch a jamming attack, that gives Eve the opportunity to receive packet and hash value that is not received by the other nodes. This hash key may be used by Eve, but Eve may only recover hash key that is disclosed before or during the jamming attack, this reduce the effect of the attack.

If Eve, is staying in one location for a time to record message and protocol execution, and later move to another location. Eve have the opportunity to impersonate node A (and its neighbour), since Eve can reconstruct a large part of the master hash chain (Eve

also know the certificate and the details since this is public). This is critical, especial when node do not have synchronic clock, then other nodes does not know what master hash chain key is a valid one or is a replay. In this case Eve may use the reconstructed hash chain in another location and impersonated Alice.

Wormhole attacks

Eve may establish a wormhole, if she tunnel message or protocol execution to another location in the network, out of radio coverage of the originator and its neighbour. To establish the wormhole, Eve may collaborate with other node or Eve control more than one node, which is used to tunnel message or protocol execution.

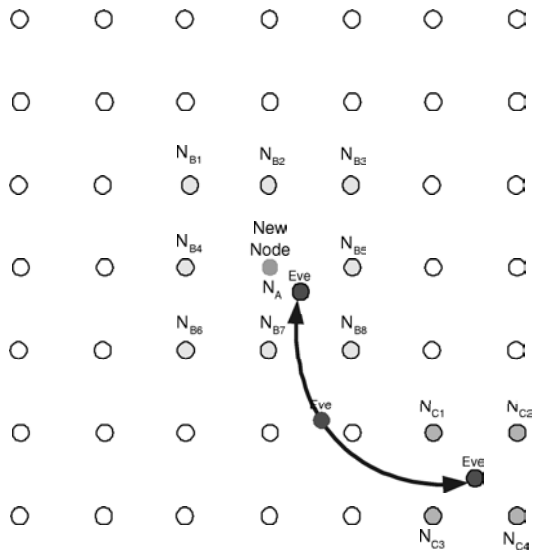

Figure 10: Wormhole attack

This is analysed as follows according phase 2 in algorithm 10 and figure 10:

- Eve forward message 1 from node A to node C (N_C), node C do not see anything wrong with this message, and recognise Node A as its neighbour.
- Node C send message 2 with a random number n_B to tell Eve how much of the second hash chain that should be disclosed. If Eve returns this message to node A; node A will recognise N_C as its neighbour, and returning message 3 according to the rules. Node C will also do send message 4 to node A.
- If there is not any wrong with node A or C, they will be authenticated.
- Then node A start to send messages to another node in the network, and immediately after sending the message node A also discloser the key. Eve can tunnel this message to node C with some small time delay, Eve may also delay the message so it receive the key and can manipulate the message and re-compute a new MAC. If the time delay is small, Node C will not be able to recognise that message is manipulated.

Eve may only tunnel message that is send, manipulates packet and increases the size of this message, but may not inject extra packet. This is also the same problem/attack that was discovered in paper [19, 42].

Denial of Service attack

In this case, Eve is trying to launch a DoS attack on Node B (N_B), and impersonate node A.

- 1. Eve generate a fake certificate and generate some value of master hash chain (h_A^M) and traffic hash chain (h_A^T). This is included in the first message (the join message). The certificate and master and traffic hash chain may belong to another node that is already disclosed.
- 2. Node B (N_B) controls the certificate is not expired and is issued from an valid TTP, that the master hash chain is valid and the last used master hash chain key is according to the first master hash chain that is signed. But node B does not verify the certificate and the signed master hash chain, before later. As a responds to the join message, node B responds with message 2 and the random number n_B (that tells how much of the traffic hash chain node A have to disclose as a evidence).
- 3. If the random number n_B is smaller then the number of key in the second hash chain then Eve has been calculated. Eve disclose the part of the traffic hash chain as expected from node B in message 3, and disclosure the traffic and master hash chain key in message 5, to authenticate message 3, immediately after message 3 is sent.
- 4. Node B controls message 3 against message 5, does the hash chain calculation, and can not se anything wrong. In this case node B does the verification of the certificate and the signed first master hash chain key.

In this case Eve have forge node B to do a verification of the false certificate and the signature of the master hash chain. This attack may be launched multiple times with different certificate, but with the same hash chain and it do not require lot of computation. But as mention earlier, to do a verification of a certificate and a signature is time consuming. According to table 1, to do the hash chain calculation does not require a lot of computation, so there must be another approach to deal with the DoS attack.

5.4.3 Phase 3: Maintenance authentication and message authentication

After a new node has joined the neighbourhood, it use message 7-10 to transmit messages through the network. In the hop-by-hop authentication protocol, messages are only broadcasted to nodes that are within radio coverage (the neighbour node), for each hop until the messages reach the destination node. The destination node verifies that the message is according to the MAC and key disclosed by its neighbour node.

Delay attack

Since every neighbour node is within radio coverage of node A, there is not possible that Eve may delay an ongoing protocol execution between node A and its neighbour. Eve may try to replay earlier message to impersonation node A or some of its neighbour, but may only replay hash key that's already have been disclosed and known by the neighbourhood.

This is the same as in phase 2, but there is a difference. This may occur any where in the network between the source and destination node. If one node is compromised this node may delay the received message, and manipulate the message before it is forwarded

to the destination node. In this case the destination node is not able to decide that the message is manipulated or not.

Replay attack

Eve may inject invalid packet into the network or launch a jamming attack in a way that nodes is dropping all received packet, which gives Eve the opportunity to receive packet and hash value that is not received by the other nodes. This hash key may be used by Eve, but Eve may only recover hash key that is disclosed before or during the attack, which reduce the effect of the attack.

This is the same problem as in phase 2, but may occur any place on the path between source and destination node.

Wormhole attacks

If a wormhole is established in phase 2 or 4, Eve may delay the message and the hash chain key, and manipulate the message (change the information and/or extend the message), and replay it into the network. In this case no node is able to detect that the message is manipulated. As mention in phase 2, Eve may only extend or change message that is already sent, and may not inject new message into the network.

It's only possible to establish new wormhole in protocol phase 2 and 4. If Eve tries to establish a new wormhole after protocol phase 2 or 4 this will be recognised, since the previous hash chain key is not known, this will be recognised during key up-date and when the traffic key is disclosed. But Eve may force node to execute phase 2 or 4, in this way Eve may always establish new wormhole.

Denial of Service attack

If Eve injects a packet with a MAC and a fault key, to force a neighbour node to do many hash operation to verify the key is according to previous disclosed keys. This kind of attack will be discovered, since the key up-date message periodically. Since this protocol phase, only includes verification of hash chain key and MAC is according to last received hash chain key and the message. There is hard to believe that this phase is vulnerable to DoS attack.

5.4.4 Phase 4: Re-authentication and change traffic hash chain

The key-update is also used when nodes is re-joining, in this way there is no need to do a verification of a certificate and a signature. But if node does not have a synchronic clock, there is not possible to distinguish between valid or not valid master hash chains key. The use of master hash chain in this case is not secure.

Delay attack

As mention earlier this protocol is only dealing with authentication of neighbour node, where there is nearly impossible for an adversary (Eve) to delay an ongoing protocol execution. One possibility is as mention in phase 2 of the protocol, when Eve is disturbing the neighbour node in a way they do not receive message from the originator. In this phase, the only Eve may gain is that nodes is not able to finish re-authentication or change traffic hash chain.

Replay attack

See the description in phase 2.

Wormhole attacks

See the description in phase 2.

Denial of Service attack

Eve may try to cheat the neighbour node to do a lot of hash chain calculation. If Eve has impersonate Alice (by use Alice ID), and inject a packet with a hash value that do not belong to Alice. The neighbour node may have to do a number of hash operations when it tries to verify the packet and hash key is according to last received key from Alice. But since the key up-date is periodical, it reduce the number of hash operation. This kind of attack may also happen in all phases of the authentication protocol.

5.4.5 Conclusion

The most critical in this protocol is that all nodes must have synchronised clock, to be able to know that the received master hash key is fresh and is not a replay. The other problem is that, it is possibility to launch a DoS attack in phase 2. Before the responder do the verification of initiators signature and certificate, the initiator has to prove that he has noble intentions. It is also possible that an adversary will force a responder to do lot of hash computation, when the responder is verifying disclose of key. In addition the wormhole attack has to be considered, since this is a real treat to the MANET.

Another threat is if one node is compromised, then it may also compromise the other node that it has been communicated with. Since it store the last value of the master hash chain. If every node have an accurate clock this, attack will not be possible. A compromised node may also acts as an adversary to delay, manipulate, replay or inject packet on the path between source and destination node.

5.5 Review of authentication protocol and counter measures

5.5.1 General

In the security analysis of the hop-by-hop authentication protocol, several security faults are identified, and the protocol have to be reviewed to meet this security weaknesses.

5.5.2 Freshness to counter replay attack

According to the security analyse, the master hash chain key is vulnerable to replay attack. To protect against this attack, the protocol has to guarantee that the key always is fresh. According to [25], there is two ways to guarantee this:

1. use a random number or nonce as a challenge
2. use time stamp to guarantee that the key is fresh

The first choice requires public/private key operation according to algorithm 14, 15, 16 and 17 in the annex, this must be done every time when authentication is needed. This approach is vulnerable to DoS attack, and is not computation efficient. The second choice may also be done bye using public/private key operation as in algorithm 13 in the Annex. But if every node have a synchronised clock with a small deviation, and each master hash chain key has a time stamp or is related to a defined time period when it is disclosed, every node may verify what time period the master hash chain belongs to.

According to [23], it is hard to obtain time synchronisation within a MANET. But as described in section 4.6 with the development of small GPS-chip and it is more common to include this function in smaller mobile device. It seems to be possible to include this

kind of GPS-receiver (a one chip circuit) in a node that can be used in MANET, and all nodes have synchronised clock.

Time control of master hash chain key

If we expect that every node disclosure a key from the master hash chain every second from that time the hash chain is valid, and that the key is used one time. Then every key has a time frame.

According to table 3, doing a hash operation and generating a hash chain is a fast operation. With the assumption that every nodes have a synchronised clock with a small deviation, it should be possible to generate a hash chain where each hash chain key is related to defined time period. One option is that each key have a time stamp, where the time stamp may be used to prove that the disclosed hash chain key is fresh. One way to time stamp the hash chain key is given in figure 11, when the hash chain is generated every hash chain key is stamped with the time it will be disclosed. Then the receiver is able to verify that the hash chain key is fresh and belongs to a given period of time, and belongs to the initiator. If the hash chain key have a time stamp, the receiver also knows how many time it have to apply the hash function, to the result from the XOR operation to the key and the actual time for the key, to verify that the disclosed key belongs to the same hash chain that is authenticated and signed by the initiator. Another option is to disclose master hash key at each time interval, in this way there is no need of keys with time stamp. In this option, the receiver also know how many time it have to apply the hash function, to verify that the disclosed key belongs to the same hash chain that is already authenticated, and belongs to the initiator.

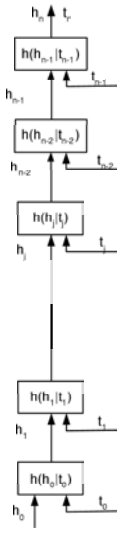

Figure 11: Generation of hash chain and keys with time stamp

5.5.3 Protect against wormhole attack

To protect against this kind of attack, the first option is to protect an attacker to be able to establish a wormhole. The wormhole has to be established in phase 2 or 4 of the protocol, and can not be established in phase 3, as it can be detected. To protect against wormhole attack, node have to be able to detect that the message is not from the neighbourhood. The second option is to detect in phase 3 that the message has been manipulated, and is

not authentic when it is received at destination node.

Protect against establishing of wormhole

This option requires the node doing authentication has to be assured that they is within radio coverage of each other. This may be done by controlling the position to each other, or measure the delay of message.

The position to a node may be defined by signal strength or GPS-pos. Since a node is mobile and the signal strength will be fluctuation in the wireless environment, and adjusted to reduce the power consumption. The antenna may have gain depending on direction. To measure the position to a node, nodes have to collaborate to compute the position to the node [35]. The collaborating node have also to know each other position, to be able to compute the position to the n GPS position depends on every node have nearly continues connection with at least 4 GPS-satellites. Both of this solution may not be stabile and reliable.
Another problem with including nodes position is that the position has to be signed. If not, the adversary is able to delay the message until the key is disclosed, and change the position and re-compute the MAC.

To detect a delayed message, nodes have to be tight time synchronic. If the radio coverage is 100 meter, and the electromagnetic waves that carry the message is travelling with speed of light (approximately $3*10^8[m/s]$). Then it have to be possible to detect that the message has travelled more than 200 meter, that's the same as $0,67$ microsecond in delay. The use of GPS make this possible, but it requires that every node have continues connection with at least 4 GPS-satellite, since the limit is 340 nanosecond according to [30]. It seems impossible to obtain this tight time synchronisation between nodes, and it will also require a lot of power to be able to continually do the synchronisation of clock according to GPS-satellite.

Detect that message has been manipulated

Another approach is to be able to detect that the message is manipulated between source and destination node. This will not protect against establishing of wormhole, but to be able to detect manipulated message in phase 3. The approach requires establish an authenticated channel between source and destination node, and an end-to-end authentication protocol between source and destination node.

5.5.4 End-to-end authentication

As mention earlier, the hop-by-hop protocol is not suited to be used as end-to-end authentication, since the message and the late disclosure of the key may follow different path. This may give different time delay and an adversary node (Eve) may delay the message until the disclosure of the key as in figure 12. In this way, Eve may change the message and re-compute the MAC without that the receiver notices the changes. In the finale authentication protocol this has to be considered.

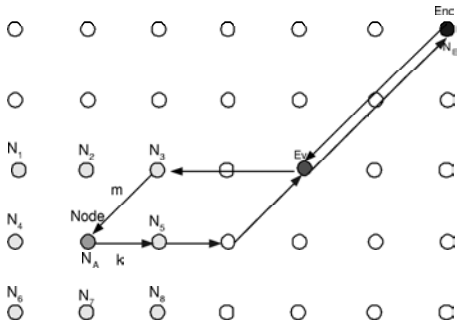

Figure 12: End-to-end authentication

6 New authentication protocol

In this chapter and figure 13, the final proposed authentication protocol is presented. The authentication protocols consist of two parts, one part is a hop-by-hop authentication protocol and the second part is an end-to-end authentication protocol.

As mention in the security analyse and review of the first proposed authentication protocol in chapter 5, node must have a synchronised clock with small deviation when the master hash chain is used instead of public/private key operation. In the new authentication protocol, it is assumed that every node have a GPS-receiver and synchronic clock, which is synchronic according to the GPS. With timely clock (with a small deviation much less than a second between nodes), there is no need to negotiate or determine the other nodes' clock, or the time difference between nodes by protocol execution.

The hop-by-hop authentication protocol is to secure that only legitimate node is part of the network, and protect the message between neighbour node according to figure 13. The protocol covers only authentication between neighbour nodes, and can not be used as authentication between a source and a destination node, where it requires multiple hop in between.

The end-to-end protocol is to secure that a source and a destination node is authenticated and the received message is authenticated. If the messages have been manipulated along the path between the source and the destination node, the destination node is able to detect this.

Both protocol parts (hop-by-hop and end-to-end authentication protocol) are based on two hash chain (which is to be generated and stored, and used as authentication key) and public/private key crypto algorithm. The first hash chains are the master hash chain which is protected by the owner's signature, and is used to reduce the need of public/private key operation, and maintenance authentication during a time period or session. The second hash chain is used as traffic key or session key, which is used to authenticate message among neighbour node or source and destination node. If there is necessary to change master hash chain, the new one has to be signed with the owners' private key. As such, other node is able to verify that this master hash chain belongs to the claimed owner.

The same master hash chain is used in both part of the protocol in the same way. The traffic hash chain is different. In the hop-by-hop authentication protocol, nodes use the traffic hash chain key to authenticate broadcast message between neighbour nodes. If there is necessary to change the traffic hash chain, this is done by initiating phase 4 of the protocol, and should be done at least one key up-date period before the change will be executed. In the end-to-end protocol, the traffic hash chain is replaced by a number of session hash chain, which is related to each session between the source and the destina-

tion node. There have to be a number of session hash chain according to the maximum number of session which can be established for a node.

The key up-date message has the same function in both protocols it is to maintenance authentication during a time period and to confirm what is the last used traffic, session key and to authenticate previous key up-date message. In this way, the last used master hash chain key, traffic and session key is authenticated. This key up-date message is authenticated by the next key up-date message. The key up-date message is periodical (in this thesis key up-date period is set to one second). To reduce the use of transmitting power, the key up-date is only transmitted during period where there is information to be exchanged between a source and a destination node. If there is no need to exchange information for a period, then there should be send one last key up-date both to its neighbour and destination node. In this way, the last message is also authenticated. If there is not possible to send the key up-date at same time to neighbour and the end nodes (end nodes is the destination node for a communication session), it first sends the key up-date to its neighbour and then to end nodes with the corresponding session traffic key. The next time a node have to send a message and the master hash chain has not been changed, they have to re-authenticate the node disclosure the master hash key that belongs to this time period and is authenticated by the next master hash chain key. The current master hash chain key is according to equation 6.1. Where n_c is the current master hash chain key, n_l is the last disclosed master hash chain key and n_p is the number of key up-date period after last transmitted key up-date message.

$$h^M_{n_c} = h^M_{n_l + n_p} \tag{6.1}$$

More details about the protocol are described in the following section.

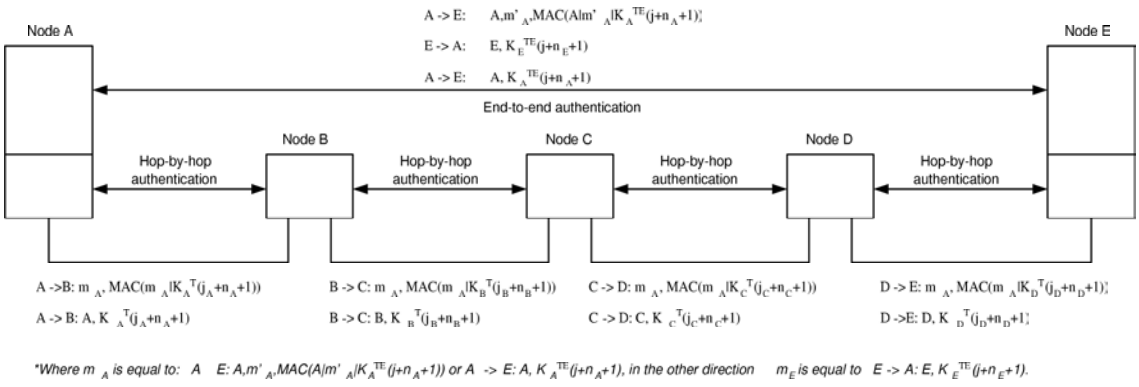

Figure 13: Message authentication scheme

6.1 Description of hop-by-hop authentication protocol

As mention in the security analyse in section 5.4, node must have a synchronised clock with small deviation (it is assumed that every node have a GPS receiver). With timely clock, every neighbour to a node knows that the latest disclosed key (from the master hash chain) is fresh, received timely, and is not a replay of recorded key from another location in the network. But it does not protect against wormhole attack, which require

that the deviation is smaller than a microsecond.

To handle the possibility of DoS attack, there must be some adjustment to the protocol. There have to be another challenge responds method, then was used in algorithm 10 in chapter 5. The initiator has to prove that he has noble intention to complete the authentications process. In the finale hop-to-hop protocol the initiator has to compute a word or number. If the protocol includes a random number r_A in message 1, and this is used by Node B to return a message that include a hash of r_A and r_B, where r_B is a random number chosen by node B. Node B expects that Node A solve the hash function and return r_B. In this way, node A proves that it really wants to talk to node B. For node B, to validate that node A has done the computation of r_B do not require heavy computations, since it is only necessary to compare the random number that was generated r_B with the value r'_B that node A has computed. If $r_B = r'_B$, then node B knows that node A really want to talk to node B. r_B has to be reasonable large to guess, to protect against DoS attack.

As described earlier, the protocol as present in algorithm 11 consists of four phases: Phase 1: Pre-distribution of credential, Phase 2: Bootstrap process and change master hash chain, Phase 3: Maintenance of authentication and message authentication and Phase 4: Re-authentication and change traffic hash chain.

Algorithm 11 Hop-by-hop Authentication protocol Final

Phase 2 Boot-strap and change master hash chain:

1: $A \rightarrow B : Cert_A, Sign_A(A|K_A^M(0)|T_A^M(0)),$
$r_A, K_A^M(i_A - 1), K_A^T(j_A), MAC(r_A|K_A^M(i_A)|K_A^T(j_A))$

2: $B \rightarrow A : Cert_B, Sign_B(B|K_B^M(0)|T_B^M(0)),$
$h(X_B|r_A), K_B^M(i_B - 1), K_B^T(j_B), MAC(h(X_B|r_A)|K_B^M(i_B)|K_B^T(j_B))$

3: $A \rightarrow B : A, B, h(X_B), MAC(A|B|h(X_B)|K_A^T(j_A + 1))$

4: $A \rightarrow B : A, K_A^T(j_A + 1)$

Key up-date:

5: $A \rightarrow B : A, K_A^M(i_A), K_A^T(j_A + 1), MAC(A|K_A^M(i_A + 1)|K_A^T(j_A + 1))$

6: $B \rightarrow A : B, K_B^M(i_B), K_B^T(j_B), MAC(B|K_B^M(i_B + 1)|K_B^T(j_B))$

Phase 3: Maintenance authentication and message authentication

7: $A \rightarrow B : A, m_A, MAC(A|m_A|K_A^T(j_A))$

8: $A \rightarrow B : A, K_A^T(j_A)$

9: $A \rightarrow B : A, K_A^M(i_A + 1), K_A^T(j_A), MAC(A|K_A^M(i_A + 2)|K_A^T(j_A))$

10: $B \rightarrow A : B, K_B^M(i_B + 1), K_B^T(j_B), MAC(B|K_B^M(i_B + 2)|K_B^T(j_B))$

Phase 4: Re-authentication and change traffic hash chain

11: $A \rightarrow B : A, K_A^M(i_A), K_A^T(0), MAC(A|K_A^M(i_A + 1)|K_A^T(0))$

12: $B \rightarrow A : B, K_B^T(j_B), MAC(B|K_B^M(i_B + 1)|K_B^T(j_B)$

Key up-date:

13: $A \rightarrow B : A, K_A^M(i_A + 1), K_A^T(0), MAC(A|K_A^M(i_A + 2)|K_A^T(0))$

14: $B \rightarrow A : B, K_B^M(i_B + 1), K_B^T(j_B), MAC(B|K_B^M(i_B + 2)|K_B^T(j_B)$

6.1.1 Phase 1: Pre-distribution of credential

Before node is deployed, nodes get the certificate with a unique identity and public key from a TTP, the corresponding private key is stored in tamper proof memory or device within the node. The node can start to generate the hash chain K_A^M. When the generation

of the hash chain is finished, the first mater hash chain key $K_A^M(0)$ and valid time period $T_A^T(0))$ is signed by node A's private key. This has to be done before the node join the network. The traffic hash chain (K_A^T) is generated, to authenticate message exchange with neighbour nodes.

6.1.2 Phase 2: Bootstrap process and change master hash chain

In this phase, the node proves its identity. All node have to go through this phase the first time a node is proving its identity to another node that is not already authenticated, or when the node have to change the master hash chain.

Message 1

Node A broadcasts the message to its neighbour node, that contain A's identity and certificate. It also includes the first master hash chain key $K_A^M(0)$, the valid time period $T_A^M(0)$ (the start and end time) for the master hash chain, identity to node A (A) and node A's signature. Where the signature includes $K_A^M(0)$, $T_A^M(0)$ and nodes identity, which is signed with nodes A private key. Message 1 also includes, the last used master key K_A^M, traffic key K_A^T, a random number r_A, and a MAC that is computed based on the next master hash chain key $(K_A^M(i_A)$. The random number r_A shows that the node have the motivation to compute the challenge from node B, where $r_A < 160bit$. The size to r_A will be discussed later in the security analyses.

Message 2

Message 2, is a response to message 1 and includes the same information as in message 1. Node B, verifies that the certificate from node A is not expired and valid, the valid time period of the master hash chain $T_A^M(0)$, and that $K_A^M(i_A - 1)$ is according to the first master hash chain key $K_A^M(0)$, and that $r_A < 160bit$. It also includes an challenge $h(X_B|r_A)$, where node A have to compute X_B and return $h(X_B)$.

If Node A received more than one responds (message 2), it stores them and answer them later. Node A validate node B's responds (message 2), that the certificate is not expired and valid, the valid time period of the master hash chain $T_B^M(0)$, $K_B^M(i_B - 1)$ according to the first master hash chain key $K_B^M(0)$, and $h(X_B|r_B) = 160$ bit. Node A compute X_B on basis of $h(X_B|r_A)$ and r_A.

Message 3 and 4

In this message, node A proves that it has solved the challenge by including $h(X_B)$ in message 3. Since an adversary may have more computation power than node A, and is able to compute the challenge before node A. To assure that the message 3 is from node A, the message has to be authenticated. Therefore the message has an MAC that is authenticated by message 4. After node A has sent message 3, it discloses the next traffic key in message 4, immediately after message 3.

Key up-date

Node A and B transmit a key up-date at regularly key up-date time, to authenticated disclosed traffic key and master key.

Verification of nodes

After receiving the key up-date, node A and B are able to verify each others message.

1. Node B verify that node A has solved X_B.

2. Node A and B verifies that the key up-date message (message 5 and 6) is according to value in message in message 1 and 2.
3. Node B verifies that the disclosed key in message 4 is according to message 5.
4. Node B verifies that the MAC in message 3 is according to key in message 4.
5. If all verification in 1-4 are correct, node A and B verify the signature and the certificate in message 1 and 2.
6. If all item 1-5 are correct, then the node is authenticated.

6.1.3 Phase 3: Maintenance authentication and message authentication

The first part must be efficient to transporting data through the network, without time consuming computation, to authentication the message at every node on the path between source and destination node. This is also the reasons the protocol is based on MAC and hash chain, since they are nearly or as efficient as symmetric algorithm according to table 3.

Message 7

Is used to exchange messages between neighbour nodes. Where the message is authenticated by a MAC, which includes the identity, message m_A and the next traffic key K_A^T as input parameters.

Message 8

After message 7 is transmitted, the next traffic key K_A^T is immediately disclosed in message 8 to authenticate message 7.

Message 9 and 10 Key up-date

The key up-date (message 9 and 10) is periodical distributed within the neighbourhood, to maintenance authentication and authenticate traffic keys. This key up-date message is authenticated by the next key up-date message. To save energy node should only transmit key up-date messages when it has something to send or route. In this case, the last key up-date message is sent to its neighbour one key up-date period after the last message.

6.1.4 Phase 4: Re-authentication and change traffic hash chain

According to message 9 and 10 in phase 3, there are possible that nodes within the neighbourhood do not communicate continuous. Node may also re-join previous authenticate node, if the node or other nodes have left this part of the network for a time period. Then they have to re-authenticate. If a node after a period of time, have to send a message to its neighbourhood to nodes that is already authenticated. They have to re-authenticate, to tell the neighbourhood what is the last used traffic key and the current master hash chain key (which belong to this time period). This is also the case if a node has to change the traffic hash chain. The new traffic hash chain has to be committed and authenticated, before the neighbourhood may trust this hash chain. In this protocol, the new traffic hash chain is committed (signed by the master hash chain key) and authenticated by the master hash chain key. In this way, private/public key operation is only need the first time nodes are authenticating each other, or when the master hash chain have been changed.

Message 11-14

If the node has to change its traffic hash chain, it has to send message 11 to its neighbourhood, and look for responds as in message 12. The new traffic hash chain is authenticated

by the next key up-date message 13.

Key up-date

This message is also used when nodes are re-joining after a period of time. In this case nodes are already been authenticated in phase 2. But if the master hash chain has been changed, they have to go through phase 2 to commit and authenticate the new hash chain.

6.2 Description of end-to-end authentication protocol

The hop-by-hop authentication protocol do not support end-to-end authentication, since it only deals with authentication of message among neighbour nodes. With some modification, it is possible to use the same approach as in the hop-by-hop authentication. The major difference is that the late disclosure of the hash chain value, must be controlled by an acknowledge message from the destination node. Otherwise, late disclosure of the key, may give different time delay when the message and disclosed key propagated through the network. And a compromised node may delay a message until the authentication key is disclosed or received. If so happen there is a security break in the protocol, since the compromised node may change the message and compute a new MAC. In this case, the destination node is not able to detect that the message and MAC has been changed.

The protocol includes two hash chain a master hash chain K^M (the same as in the hop-by-hop authentication protocol), and a session hash chain (and K^S) for each session a node may establish.

The end-to-end authentication protocol is presented in algorithm 12, which consist of four phases: Phase 1: Pre-distribution of credential, Phase 2: Bootstrap process and change master hash chain, Phase 3: Maintenance authentication and message authentication and Phase 4: Re-authentication and change session hash chain.

In this protocol node A is the initiator and node B is the destination node. Between node A and B, there may be a number of intermediate node, which route the message from the initiator to destination node.

6.2.1 Phase 1: Pre-distribution of credential

Phase 1 in the end-to-end protocol, is the same as in hop-by-hop protocol.

6.2.2 Phase 2: Bootstrap process and change master hash chain

In this phase, the node proves its identity and commits the master hash chain. All node have to go through this phase the first time a node is proving its identity to another node that is not already authenticated, or when the node have to change the master hash chain.

Message 1

Node A send the message to the destination node, that contain A's identity and certificate. It also includes the first master hash chain key $K_A^M(0)$, the valid time period $T_A^M(0)$ (the start and end time) for the master hash chain, node A's identity and a signature. Where the signature includes $K_A^M(0)$, $T_A^M(0)$ and node A's identity, which is signed by node

Algorithm 12 End-to-end authentication protocol final

Phase 2 Boot-strap and change master hash chain:

1: $A \rightarrow B : Cert_A, Sign_A(A|K_A^M(0)|T_A^M(0)),$
$r_A, K_A^M(i_A - 1), K_A^S(j_A), MAC(r_A|K_A^M(i_A)|K_A^S(j_A))$

2: $B \rightarrow A : Cert_B, Sign_B(B|K_B^M(0)|T_B^M(0)),$
$h(X_B|r_A), K_B^M(i_B - 1), K_B^S(j_B), MAC(h(X_B|r_A)|K_B^M(i_B)|K_B^S(j_B))$

3: $A \rightarrow B : A, B, h(X_B), MAC(A|B|h(X_B)|K_A^S(j_A + 1))$

4: $B \rightarrow A : B, K_B^S(j_B + 1)$

5: $A \rightarrow B : A, K_A^S(j_B + 1)$

Key up-date:

6: $A \rightarrow B : A, K_A^M(i_A), K_A^S(j_A + 1), MAC(A|K_A^M(i_A + 1)|K_A^S(j_A + 1))$

7: $B \rightarrow A : B, K_B^M(i_B), K_B^S(j_B + 1), MAC(B|K_B^M(i_B + 1)|K_B^S(j_B + 1))$

Phase 3: Maintenance authentication and message authentication

8: $A \rightarrow B : A, m_A, MAC(A|m_A|K_A^S(j_A + 2))$

9: $B \rightarrow A : B, K_B^S(j_B + 2)$

10: $A \rightarrow B : A, K_A^S(j_A + 2)$

Key up-date:

11:$A \rightarrow B : A, K_A^M(i_A + 1), K_A^S(j_A + 2), MAC(A|K_A^M(i_A + 2)|K_A^S(j_A + 2))$

12:$B \rightarrow A : B, K_B^M(i_B + 1), K_B^S(j_B + 2), MAC(B|K_B^M(i_B + 2)|K_B^S(j_B + 2))$

Phase 4: Re-authentication and change session hash chain

13:$A \rightarrow B : A, K_A^M(i_A), K_A^S(0), MAC(A|K_A^M(i_A + 1)|K_A^S(0))$

14:$B \rightarrow A : B, K_B^M(i_B), K_B^S(0), MAC(B|K_B^M(i_B + 1)|K_B^S(0))$

Key up-date:

15:$A \rightarrow B : A, K_A^M(i_A + 1), K_A^S(0), MAC(A|K_A^M(i_A + 2)|K_A^S(0))$

16:$B \rightarrow A : B, K_B^M(i_B + 1), K_B^S(0), MAC(B|K_B^M(i_B + 2)|K_B^S(0))$

A's private key. Message 1 also includes, the last used master key K_A^M, session key K_A^S, a random number r_A, and a MAC that is computed based on the next master hash chain key ($K_A^M(i_A)$. The random number r_A shows that the node have the motivation to compute the challenge from node B, where $r_A < 160bit$. The size to r_A will be discussed later in the security analyses.

Message 2

Message 2, is a response to message 1 and includes the same information as in message 1. Node B, verifies that the certificate from node A is not expired and valid, the valid time period of the master hash chain $T_A^M(0)$, and that $K_A^M(i_A - 1)$ is according to the first master hash chain key $K_A^M(0)$, and that $r_A < 160bit$. It also includes an challenge $h(X_B|r_A)$, where node A have to compute X_B and return $h(X_B)$.

In the end-to-end protocol the initiator expect only one responds for each session that is being established. Node A validate node B's responds (message 2), that the certificate is not expired and valid, the valid time period of the master hash chain $T_B^M(0)$, $K_B^M(i_B - 1)$ according to the first master hash chain key $K_B^M(0)$, and $h(X_B|r_A) = 160$ bit. Node A compute X_B on basis of $h(X_B|r_A)$ and r_A.

Message 3

In this message, node A proves that it has solved the challenge by including $h(X_B)$ in message 3. Since an adversary may have more computation power than node A, and is able to compute the challenge before node A. To assure that the message 3 is from node A, the message has to be authenticated. Therefore the message has an MAC that is authenticated by message 5.

Message 4

Before node A discloser the session key that is used to authenticate message nr 3, node A expect a responds from Node B. When node B, has received message 3 and have validated the respond to the challenge $h(X_B|r_A)$, and node A have computed $h(X_B)$. Then Node B responds with acknowledge, which is a disclosure of the next session key and can be verified by node A.

Message 5

When node A receive message 4, it verify that the session key is according to the previous key in message 2. If the key is valid, node A disclosure the session key to authenticate message 3. In this way, there are not possible to delay the message (message 3) until the session key is disclosed. And an adversary or compromised node is not able to change the message, without the destination node is able to detect that the messages is changed.

Message 6-7 Key up-date

Node A and B transmit a key up-date at regularly key up-date time, to authenticated disclosed session key and master key.

Verification of nodes

After receiving the key up-date, node A and B are able to verify each others message.

1. Node B verify that node A has solved X_B
2. Node A and B verifies that the key up-date message (message 6 and 7) is according to keys in message in message 1 and 2.

3. Node B verifies that the disclosed key in message 5 is according to message 1 and 6.
4. Node B verifies that message 3 is authenticated by the session key in message 5.
5. Node A verifies that the responds in message 4, is according to session key in message 2 and 7.
6. If all verification in 1-5 are correct, node A and B verify the signature and the certificate in message 1 and 2.
7. If all item 1-6 are correct, then the node is authenticated.

6.2.3 Phase 3: Maintenance authentication and message authentication

Message 8

The first part must be efficient without time consuming computation to authenticate the message at the destination node, and for the initiator to generate the value that is used to authenticate the message. This is the reasons the protocol is based on MAC and hash chain, since they are nearly or as efficient as symmetric algorithm according to table 3.

Message 9

Before node A disclose the session key, which is used to authenticate message 8. Node A expects a responds from node B. The responds from node B is the next session key, which can be verified by node A.

Message 10

When node A receive the responds from node B (message 9), the disclosed session key from node B, is validate according to earlier received session key. If the validation succeeds, node A discloses the session key to authenticate message 8.

In this way, there are not possible to delay the message (message 3) until the session key is disclosed. And an adversary or compromised node is not able to change the message, without destination node may detect that the message is changed.

Message 11 and 12 Key up-date

The key up-date (message 11 and 12) is periodical sent to node that belongs to a session, to maintenance authentication and authenticate session keys. This key up-date message is authenticated by the next key up-date message. To save energy node should only transmit key up-date messages when it has something to send. In this case, the last key up-date message is sent one key up-date period after the last message.

6.2.4 Phase 4: Re-authentication and change session hash chain

According to message 11 and 12 in phase 3, there is possible that nodes related to a session do not communicates continuous. If they after a time period have to exchange message, they is able to use the same credential that was used during first time they authenticated. But they have to re-authenticate, to tell the neighbourhood what is the last used session key and the current master hash chain key. This is also the case if a node has to change the session hash chain. The new session hash chain has to be committed and authenticated, before the other node may trust this hash chain. In this protocol, the new session hash chain is committed (signed by the master hash chain key) and authenticated by the master hash chain key.
In this way, private/public key operation is only need the first time nodes are authenticating each other, or when the master hash chain have been changed.

Message 13-14

If the node has to change the session hash chain, it has to send message 13 to the destination node, and look for responds as in message 14. The new session hash chain is authenticated by the next key up-date message 15.

Key up-date

This message is also used when nodes are re-joining after a period of time. In this case nodes are already been authenticated in phase 2. But if the master hash chain has been changed, they have to go through phase 2 to commit and authenticate the new hash chain.

6.3 Security

This section covers the security analysis on the new authentication protocol, both the hop-by-hop authentication protocol and end-to-end protocol.

Since the new authentication protocols are based on late key disclosure, it is important that every node has control on the last used, current and next key. And it is not allowed to use a hash key as an MAC key, when it is disclosed. All use of hash chain key (master, traffic and session) in these protocol, have to follow this principles.

One possible attacks, is that an adversary may try to force a responder to do a number of hash operation, without the responder is able to verify that the last disclosed hash key is according last known key. Master hash chain is not vulnerable to DoS attack. Since the first master hash chain key has a time stamp and the other key belongs to a given time period. Every node that has completed the authentication of master hash chain in phase 2, is able to estimate how many hash operation that is necessary, to verify that last disclosed key is according to the first master hash chain key. In addition, when a node receives a key up-date message, it stores the last disclosed master hash chain key and the time it was disclosed. The traffic and session hash chain key does not have a time stamp. But since the key up-date is received periodic, it is possible to detect a DoS attack when the next key up-date is received.

Phase 1: Pre-distribution of credential

This phase is the same for both protocols. Where the credential is distributed over an off-line and authenticated channel, where the node (or owner) has to prove it is identity before credential is delivered and installed within the node. The TTP and the off-line authenticated channel have to be protected and secure, if not the entire network may be compromised. The private key has to be stored in a secure area within the node that is tamper proof. If a claimed owner does not have access its private key, other nodes can not trust this node. This is the fundamental of private/public and secret encryption system, where both private and secret key have to be stored in a secure area where only the owner has access. In this protocol it is assumed that no adversary has access, or is able to launch an attack in this phase.

6.3.1 Security analyse hop-by-hop authentication protocol Final

As mention earlier, this protocol is only used to authenticate neighbour node within radio coverage, where every message (protocol execution) is broadcasted to neighbour nodes.

There is two ways that an adversary may delay or delete protocol execution:

- If an adversary is able to establish a worm hole (tunnel message from one position to another position in the network).
- The other option is to disturb (by jamming or inject faulty packet) the neighbourhood, in a way that the receiving nodes are not able to receive the message.

In both case, the adversary is able to delay the message (protocol execution) until the authentication key is disclosed. Then the adversary is able to change or extend the message and compute a new MAC, before the message is broadcasted by the adversary.

If the adversary is broadcasting the message within the neighbourhood of the originator, the originator is able to detect that the message is manipulated, and may send a warning to the neighbourhood. If the adversary has established a wormhole, either the originator or the other neighbourhood (in another location) are able to detect that the message is changed or extended. But the adversary is only able to extend or change already transmitted message. If an adversary tries to inject a new packet, this will be discovered. Since it is assumed that the adversary is not able to compute the next traffic or master key, before the next key up-date message is transmitted. Then the adversary may use a false or earlier disclosed traffic/master key, which will be discovered by the neighbour node. When they verify the disclosed key according to last received key or key up-date message.

If an adversary is not able on establish a wormhole or succeed with jamming. It is hard to delay or delete protocol execution, since the message is travelling as an electro magnetic wave with speed of light, within the neighbourhood.

If an adversary tries to replay an earlier message or protocol execution within the neighbourhood, this will be detected by the neighbour node. Since it use the same traffic hash chain key, which has been disclosed earlier or that the master hash chain key belong to an earlier key up-date period. If an adversary is recording packet (messages) at a location, and later replay messages at a new location. Node at the new location is able to detect that the master hash chain key belongs to an earlier key up-date period, since each master key belongs to a period in time. Another option, if an adversary tunnel packet to another location (in near real time manner), in this case node at both location seems to be within radio coverage. If an adversary try to tunnel packet to the other location after protocol phase 2 (and 4, if there are node at the other location that earlier have been authenticated), the tunnelled packet will be detected as it includes hash chain key that is not authenticated. To succeed, the wormhole has to be established during phase 2 (or 4). In this situation, the adversary is able to delay packet (message) until the traffic key is disclosed. Then the adversary may change or extend the message and compute a new MAC, without no one is able to detect that the message is changed.

Phase 2: Bootstrap process and change master hash chain

This phase involve to replace the public/private key with a master hash chain key, which will be used to maintenance authentication during a time period and authenticate traffic key. It includes public/private key operation that may be vulnerable to DoS attack, if it is not taken in to account in the design of the protocol.

The real treat in this phase is that an adversary tries to establish a wormhole as

mention earlier. If an adversary, is able to tunnel information from node A to node B, where node B is in another location and out of radio coverage of node A. It is possible to complete the authentication of node A and B. But the adversary is not able to impersonate node A or B. If Eve tries to change the certificate or the signed master hash chain, the other node can detected this as it verifying the certificate and the signature according to the public key. If the current master hash chain key is manipulated or is replayed from earlier disclosed key, this can be detected since nodes is able to verify that the key does not belongs to the actual hash chain or is not the current key.

If one or more messages are manipulated or missing during phase 2, every node within the neighbourhood is able to detect this. Since every message is linked together by the traffic key and master hash chain key. If one of the neighbour nodes does not receive the key up-date message, it may wait until the next key up-date is received. But if at least one of messages 2-4 is missing, the authentication has to be terminated, since node within the neighbourhood does not have necessary information to complete the authentication.

If an adversary (in this case node A), try to launch a DoS attack to node within the neighbourhood, by forcing neighbour node to do heavy computation as verification of certificate or signature. If the adversary try to launch this attack, it have to prove that it really want to talk to node B (a number of node within the neighbourhood) by compute the challenge X_B. To balance the protocol, X_B should be chosen to require more computation than doing the verification of certificate and signature. But require less computation than generate a signature. As in the protocol, the initiator has to prove that he has computed the challenge, before the responder has to verify initiators certificate and signature. It is assumed, that an adversary have no intension to compute the challenge, since it require more computation than verifying node A signature, and have to use more power than the responder (in this case node B).

Phase 3: Maintenance authentication and message authentication

If Eve (the adversary), has established a wormhole in phase 2 (or 4). Eve is able to tunnel every message from one location to another location in the network. Where Eve may delay messages until the traffic key is disclosed. In this case, Eve is able to change or extend the message and compute a new and MAC, since it has access to the traffic key. Eve may send the manipulated message to the other location, and disclose the traffic key immediately in an ordinary way. In this case the receiving node is not able to detect that the message is manipulated. To succeed with this attack, Eve have to delay, manipulated (change or extend) the message and compute a new MAC, before receiver node expect the next key up-date message.

Another attack is to change the traffic key, to force node B to do a number of hash operation without be able to verify the traffic key. This type of attack is possible to detect after a few hash computation, since the last used traffic key authenticated by the next key up-date message which is transmitted periodical.

If an adversary tries to establish a wormhole during this phase, the legitimate node in both locations is able to detect, since the first master hash chain key is unknown or is not up-dated.

If there is an insider or compromised node (Eve) in the path between a source and a destination node. Eve is able to delay, manipulate or extend message without the detection

from the destination node. Eve may also inject new and replay earlier received packet into the network, and pretend to be another node (by replaying earlier message) without neighbour node or end node is able to detect that the message is changed.

Phase 4: Re-authentication and change traffic hash chain

If an adversary is able to establish a wormhole in this phase, node B that is assumed to be in another location (out of radio coverage of node A) has earlier been authenticated by node A. If no nodes in the other location, has not been authenticated by node A before. The adversary is not able to establish a wormhole in this phase. If the already exists a wormhole which has been established in phase 2, the consequence is the same as in phase 2 and 3.

Sub conclusion

To protect against wormhole and an insider, the new authentication protocol have to include an end-to-end authentication protocol. Other method to protect against this attack is considered in section 5.5, and is hard to detect when only a hop-by-hop authentication protocol is used. But this protocol is assumed to be secure against DoS attack according to the consideration in this section, that was not the case for protocol in algorithm 10 in chapter 5.

6.3.2 Security analyse End-to-end authentication protocol Final

This protocol is based on the same cryptographic function as in hop-by-hop authentication protocol, and late key disclosure. But the authentication is done between source and destination node, where there may be multiple hop in between. With a number of nodes between source and destination node, there is possible that message and disclosure of authentication key have different propagation speed. If this is not considered, it may happened that the disclosed authentication key arrive before the message. In this case, an adversary is able to manipulate (change or extend) the message and re-compute the MAC, without destination node is able to detect that the message is changed. To protect against this attack, the disclosure of authentication key have to be under control. On way to do this is used in this protocol, where the protocol requires a responds (acknowledge) from destination node before authentication key is disclosed. If the responds does not appears, the authentication key is not disclosed. To reduce the risk of losing control of the session traffic hash chain key, each source/destination pair have a session hash chain that is related to this session. When the session is terminated a last key up-date message is send at the regularly time, and then the session hash chain is deleted.

In this protocol, the destination node is able to detect that the message is changed. If an adversary launch a wormhole attack or from a compromised node and manipulate the message or the disclosed key this will be detected, since the message is not authentic. If a messages is replayed later this will also be detected since the traffic key is already known or the master hash chain key is already used and belongs to another time period. The worst that may happen is that source and destination node receive multiple of the same responds or message.

A DoS attack in phase two, will also be detected if the adversary does not compute X_B, since this has to be done before node B do the verification of node A's certificate and

signature.

It is possible to detect that messages is lost on the path between the source and destination node, since each message is related to one session key. Then the destination node is able to detect missing session key. If it receive a message that is expected to be authenticated by the next session key, but instead is authenticated by a later session key. Since a source node according to the protocol expect a responds to a message or a key up-date message. It is also able to detect there may be missing messages between source and destination node. But source and destination node, is not able to decide that loss is caused by erroneous wireless channel, an insider or adversary.

Phase 2: Bootstrap process and change master hash chain

In this phase of the protocol it is used a challenge that an initiator has to solve, to prove that he is not launching a DoS attack against node B (the destination node).

This protocol includes a responds from the destination, before the source disclose the authentication key. In this way an insider or adversary (also if it has established a wormhole) is not able to delay the message until the authentication key is disclosed. If the delay the responds from destination node it will effect the authentication, since the authentication process is terminated if the responds is missing. But this may also be caused by the erroneous wireless channel.

All messages are tied together with the session and master hash chain. Where the first master hash chain key is signed by its owner. An adversary may not impersonate this node, only if the initial value that is used to generate the hash chain or private key is known by the adversary.

Phase 3: Maintenance authentication and message authentication

In this phase of the protocol the source require a responds from the initiator, before the authentication key (session key) is disclosed. The source is able to verify that the responds is from the destination, since it include the next session key. In this way an insider or adversary is not able to delay the message until the authentication key is disclosed. The adversary or insider may delay the responds from the destination node, in this case the source do not disclosure the key. When the key up-date message is received both the source and destination node is able to detect missing messages, since each session key belongs to a message and visa-versa.

It is expected that an insider or adversary is not able to compute the next session key, before the key up-date message is transmitted.

Phase 4: Re-authentication and change session hash chain

This is the same as in phase 4 in the end-to-end authentication protocol.

6.4 Result from simulation

The protocol that is proposed, is tested with use of program according to chapter D in the appendix. Result, consideration and description of the test program are given in following section.

6.4.1 The test program

Test program according to chapter Din the appendix, is a C-program that is based on Miracl [22] and run on Amilo Pro V3505 with 1.6 GHz Centrino Duo processor. The

Compiler is Microsoft Visual Studio 2005 Version 8.0.50727.42 (RTM.050727-4200) (VC2005) from Microsoft that is free [26]. The program includes ECDSA, RSA, SHA-1 and HMAC/SHA-1 according to [10]. The program is based on ANSI C instruction and instruction from Miracl library.

To verify that crypto algorithm is implemented correct, there have been done test to verify that it is possible to verify a signature that is generated. In addition for the HMAC/SHA-1 implementation, there have been used test vectors that is include in RFC 2202 [10]. These test result is not included in this report.

6.4.2 Result from the test program

The test program is compared with the Benchmark program that is distributed with MIRACL library. The result is present in table 6, some other test result from the Benchmark program is also presented in table 7 on the same computer with the same condition as in table 6. According to this table there is only small difference between the Benchmark program (included in Miracl library) and the test program. Since the test program includes some additional operation during signature generation, which explains the small deviation. There is also some deviation in generation of the different hash chain, which is caused by different operation that is included depended on it is a master, traffic or session hash chain. The Elliptic curve that is chosen is the "secp 160 curve", which is included in Miracl library. To be able to compare which algorithm that is best suited for the proposed protocols, there have been chosen to have the same bit security for chosen algorithm. In the test program it is used RSA with 1024 bit private key and a public key that is small (the public key is 65537), for DSA it is used $p = 1024$ bit and $q = 160$ bit, ECDSA a 160 bit curve, SHA-1 and HMAC/SHA-1 with 160 bit.

Function	*Benchmark (ms)*	*Test program (ms)*
DSA 1024/160 bit		
Setup		10,3429752
Generation		5,78
signature no precomputation	2,81	2,984
signature w. precomputation	0,58	
Verification	3,29	3,406
RSA 1024 bit		
RSA key generation		273,5
1024 bit signature	7,14	7,468
1024 bit RSA verification e=3	0,04	
1024 bit RSA verification e=65537	0,25	0,5
ECDSA 160 bit		
Key generation		0,032
signature no precomputation	7,29	7,437
signature w. precomputation	1,55	
verification	9,78	10,156

Table 6: Result from benchmark test and test of protocol

In the test program, there have been used a certificate that includes the most impor-

Function	*Benchmark (ms)*
DSA 2048/256 bit exponent	
signature no precomputation	13,61
signature w. precomputation	2,90
verification	16,29
RSA 2048 bit	
2048 bit signature	33,22
2048 bit RSA verification e=3	0,10
2048 bit RSA verification e=65537	0,74

Table 7: Other benchmark test result

Function	*A number of* interaction (ms)	*Time used for each* interaction (ms)
Master hash chain based on SHA-1 (10x10000 hash key)	1078	0,01078
Traffic hash chain (10000 hash key)	94	0,0094
Session hash chain (10 session and 1000 hash key)	109	0,0109
HMAC/SHA-1 (10.000 interaction)	500	0,05

Table 8: Result from test of SHA-1 and HMAC/SHA-1

tant field according to IEEE standard X.509 certificate standard [20] as present in table 9. The size of certificate in byte is presented in table 10, which is used in the test program. To reduce the size to the certificate the public key to TTP is not included, but assumed to be stored by each legal node.

Function
Vertion
Certificate serial nr
Algorithm ID
Algorithm type
Supported Algorithm
Valid not before
Valid not after
Algorithm
Public key
UTC time
Generalized time
Issuer
Subject
Certificate signature

Table 9: RSA Certificate

The program includes simulation on every message that is included in both protocols. The result for hop-by-hop authentication protocol is present in table 11, 13,15, 16 and 17, and for end-to-end authentication protocol is presented in table 13, 14,15, 16 and 17

According to both hop-by-hop and end-to-end authentication protocol, only message 1 and 2 is dependent on the choice of crypto graphic algorithm as RSA, DSA or ECDSA. In this case the result form generation and validation of message 3-12 is presented in

Crypto algorithm	*Certificate size in byte*
RSA	288
DSA	308
ECDSA	91

Table 10: The certificate size

table 11, 13, 13 and 14. In addition required message length and computation that has to be carried out by the initiator (node A) and the responder (node B) is presented in table 13 and 14 for message 3-14 (16 for end-to-en authentication) and according to the phase they belong to.

Message nr	*A*			*B*		
	Length in byte	Generation in ms	Verification in ms	Length in byte	Generation in ms	Verification in ms
3	48	0,766	-	-	-	0,047
4	24	<<0,01	-	-	-	0,01
5	64	0,047	-	-	-	0,047
6	-	-	0,047	64	0,047	-
7	76	0,047	-	-	-	0,047
8	24	<<0,01	-	-	-	0,01
9	64	0,047	-	-	-	0,047
10	-	-	0,047	64	0,047	-
11	64	0,047	-	-	-	0,047
12	-	-	0,047	44	0,047	-
13	64	0,047	-	-	-	0,047
14	-	-	0,047	64	0,047	-

Table 11: Generate and validate message 3-14, hop-by-hop authentication

Message nr	*A*		*B*	
	Length in byte	Computation in ms	Length in byte	Computation in ms
3-6	132	0,86	64	0,151
7-10	164	0,141	64	0,151
11-14	128	0,188	108	0,188

Table 12: Computation and message length, hop-by-hop authentication

6.4.3 Consideration on different crypto algorithm

In the authentication protocol for message 1 to 2, the verifier has to verify both the signature and the certificate to the other node. To verify the MAC (HMAC) function, the verifying node has to do the MAC (HMAC) computation. These computation is included in table 15 to 17. According to table 18, there is not easy to se that RSA should be preferred instead of DSA or ECDSA. To decide, the energy consumption has to be considered more closely. Let's expect that the transmitting power P_t is higher than the computational power P_c ($P_t > P_c$) and that $P_t = x \cdot P_c$, where r is the data rate in bit pr seconds.

Then we have the energy equation according to eq 6.2, which can be written as in eq 6.3. If we compare the energy difference when RSA and ECDSA, it may be expressed as

Message nr	*A*			*B*		
	Length in byte	Generation in ms	Verification in ms	Length in byte	Generation in ms	Verification in ms
3	48	0,766	-	-	-	0,047
4	24	<<0,01	-	-	-	0,01
5	-	-	0,01	24	<<0,01	-
6	64	0,047	-	-	-	0,047
7	-	-	0,047	64	0,047	-
8	76	0,047	-	-	-	0,047
9	-	-	0,01	24	<<0,01	-
10	24	<<0,01	-	-	-	0,01
11	64	0,047	-	-	-	0,047
12	-	-	0,047	64	0,047	-
13	64	0,047	-	-	-	0,047
14	-	-	0,047	44	0,047	-
15	64	0,047	-	-	-	0,047
16	-	-	0,047	64	0,047	-

Table 13: Generate and validate message 3-16, end-to-end authentication

Message nr	*A*		*B*	
	Length in byte	Computation in ms	Length in byte	Computation in ms
Phase 2 (3-7)	136	0,87	88	0,151
Phase 3 (8-12)	164	0,151	88	0,151
Phase 4 (13-16)	128	0,188	108	0,188

Table 14: Generate and validate message 3-16, end-to-end authentication

Message nr	*A*			*B*		
	Length in byte	Generation in ms	Verification in ms	Length in byte	Generation in ms	Verification in ms
1	539	0,047	-	-	-	1,047
2	-	-	1,047	540	0,063	-

Table 15: Generate and validate in message 1-2, with RSA

Message nr	*A*			*B*		
	Length in byte	Generation in ms	Verification in ms	Length in byte	Generation in ms	Verification in ms
1	471	0,047	-	-	-	6,859
2	-	-	6,859	472	0,063	-

Table 16: Generate and validate in message 1-2, with DSA

Message nr	*A*			*B*		
	Length in byte	Generation in ms	Verification in ms	Length in byte	Generation in ms	Verification in ms
1	254	0,047	-	-	-	20,359
2	-	-	20,359	255	0,063	-

Table 17: Generate and validate in message 1-2, with ECDSA

Algorithm	*A*		*B*		*Signature* generation
	Message length in byte	Computation in ms	Message length in byte	Computation in ms	
RSA	539	1,141	540	1,157	7,468
DSA	471	6,906	472	6,922	2,984
ECDSA	254	20,406	255	20,422	7,437

Table 18: Generate and validate message 1-2, hop-by-hop authentication protocol

in eq 6.4. The result will be either positive or negative according to eq 6.6 or 6.7. If the result is positive ECDSA is preferred, if negative then RSA is preferred since it require less energy. According to eq. 6.6 or 6.7, crypto algorithm that are preferred depend on the ration between x and the data rate r. In eq. 6.2 to 6.7, m_l is the message length in bit, and t_c is the computational time in seconds.

$$E = \frac{m_l \cdot P_t}{r} + P_c \cdot t_c \tag{6.2}$$

$$E = P_c(\frac{x}{r} m_l + t_c) \tag{6.3}$$

$$E_{RSA} - E_{ECDSA} = P_c(\frac{x}{r}(m_{l,RSA} - m_{l,ECDSA}) + (t_{c,RSA} - t_{c,ECDSA})) \tag{6.4}$$

$$\frac{x}{r} + \frac{t_{c,RSA} - t_{c,ECDSA}}{m_{l,RSA} - m_{l,ECDSA}} > 0 \text{ then use ECDSA} \tag{6.5}$$

$$\frac{x}{r} + \frac{t_{c,RSA} - t_{c,ECDSA}}{m_{l,RSA} - m_{l,ECDSA}} < 0 \text{ then use RSA} \tag{6.6}$$

In the proposed hop-by-hop authentication protocol with the result from table 18 and eq. 6.5 and 6.6, if $\frac{x}{r} < 8,44 \cdot 10^{-6}$ then RSA is preferred. If it is expected that $P_t = 5 \cdot P_c$ ($x = 5$) and the data rate is $10Mbit/s$, then $\frac{x}{r} = 5 \cdot 10^{-7}$, then RSA is preferred.

Let's do the same consideration between RSA and DSA, then we get if if $\frac{x}{r} < 1,05 \cdot 10^{-5}$ then RSA is preferred to be used.

For further consideration between different protocols, it is assumed the same condition as above, and that RSA is the preferred algorithm. The result according to the different phases for the hop-by-hop and end-to-end authentication protocol is presented in table 19 and tab:tab-6-5-2-2-2.

Phase	*A*		*B*	
	Message length in byte	Computation in ms	Message length in byte	Computation in ms
Phase 2 (1-6)	675	1,954	604	1,261
Phase 3 (7-10)	164	0,141	64	0,151
Phase 4 (11-14)	128	0,188	108	0,188

Table 19: Generate and validate message 1-14, hop-by-hop authentication with RSA

Message nr	*A*		*B*	
	Length in byte	Computation in ms	Length in byte	Computation in ms
Phase 2 (1-7)	675	1,987	628	1,261
Phase 3 (8-12)	164	0,151	88	0,151
Phase 4 (13-16)	128	0,188	108	0,188

Table 20: Generate and validate message 1-16, end-to-end authentication with RSA

6.4.4 Comparing different protocols

As mention in chapter 5 the proposed authentication protocol in this master thesis is build on protocol according to [23, 42] described in section 4.3. To compare these protocols it is necessary to consider the message length, and the computation that are required by all these protocols, or more precisely the energy cost. Only the hop-by-hop authentication protocol is considered when comparing different authentication protocol, since the other protocol does not included end-to-end authentication.

If it is expected that the length of master hash chain is l_M and traffic hash chain is l_T and that $l_M = c \cdot l_T$, and there is necessary to change traffic hash chain a number of time (n) during the life time of the master hash chain. Where the traffic hash chain, has to be committed (signed) $\frac{l_M}{N}$ times. Further it is expected that the storage is the same for all three authentication protocol. That means, it is necessary to have a memory space of $l_M \cdot l_T$ times 20 byte when using SHA-1. If it is expected the same condition, storage, time to exchange message, and the use of RSA and certificate as above we have according to table 21.

Algorithm	*A*		*B*	
	Message length in byte	Computation in ms	Message length in byte	Computation in ms
12 Phase 1-2	675	9,422	604	8,672
12 Phase 4	128	0,141	64	0,151
2	492	8,609	552	8,609
3	464	8,468	464	8,468

Table 21: Comparing three algorithms based on RSA

In algorithm 2, there is necessary to not only generate a signature on master hash chain, but also on the traffic hash chain. If it is expected that the traffic hash chain has to be changed n times during the same master hash chain, which in algorithm 2 require n more signature generation during the life time of the master hash chain comparing to algorithm 12. To change traffic hash chain in algorithm 12 require going thru phase 4 to the protocol, which only include hash and HMAC computation that is rather fast comparing to signature generation.

Algorithm 3 have only one hash chain. With the assumption these protocol shall be used under the same condition as the other algorithm 12 and 2. Then it is necessary to change the hash chain more often, than master hash chain in algorithm 12, which include signature generation.

Let's assume that the length of master hash chain is L_M and traffic hash chain is L_T, and that $L_M = a \cdot L_T$ where $a > 1$. The rate to disclose master hash chain key is r_M and

traffic hash chain key is r_T, and that $r_M = \frac{r_T}{b}$ where $b > 1$. Then the life time of a master hash chain is defined as

In this case, as mention earlier algorithm 2 require that the traffic hash chain has to be signed. Then the lifetime for master hash chain is defined as $t_M = \frac{L_M}{r_M}$ and the lifetime to the traffic hash chain is $t_T = \frac{L_T}{r_T}$. The number of time there is necessary to sign a new traffic hash chain, during the lifetime of master hash chain is according to equation 6.7

$$n = \frac{t_M}{t_T} = \frac{\frac{L_M}{r_M}}{\frac{L_T}{r_T}} = a \cdot b \tag{6.7}$$

In algorithm 3 there is only one hash chain. If it is expected the same condition as for algorithm 12. The hash chain has to be changed a number of time, comparing to the lifetime of master hash chain in algorithm 12. In algorithm 3, the life time to this single hash chain may be defined as $t_X = \frac{L_M + L_T}{r_T}$, since it consume hash chain key at the same rate as traffic hash chain in the two other protocols. The hash chain length is restricted by the same storage space that is $L_M + L_T$. Then the algorithm 3, require a number of signature on changed hash chain comparing with 12 that is defined in equation 6.8. The ration n_1 between equations 6.8 and 6.7 is defined in equation 6.9, and 6.10 is for algorithm 2 and n_2 is for algorithm 3.

$$n = \frac{t_M}{t_X} = \frac{\frac{L_M}{r_M}}{\frac{L_M + L_T}{r_T}} = \frac{ab}{1+a} = b \cdot \frac{1}{1+\frac{1}{a}} \tag{6.8}$$

$$\frac{n_1}{n_2} = \frac{ab}{b \cdot \frac{1}{1+\frac{1}{a}}} = a + 1 \tag{6.9}$$

$$n_2 = \frac{n_1}{a+1} \tag{6.10}$$

If it is assumed that the data rate is $10\frac{Mbit}{s}$ and the max size of a packet is 1024 byte, which result in there is possible to send 1220 packet during a second with maximal size on the packet. Let assume that, $L_M = 1.000.000$, $r_M = 1\frac{key}{s}$, $r_T = 1000\frac{key}{s}$, and every 100 second there is necessary to change traffic hash chain in algorithm 2 and 12. These assumptions give that, $n_1 = \frac{1.000.000}{100} = 10.000$, $a = 10$, $b = 1000$, $L_T = 100.000$ and $n_2 \approx 909$.

Algorithm	*Initiator*		
	Nr of times	Computation each time in ms	Total computation in ms
12 Phase 1-2	1	9,422	
12 Phase 4	10.000	0,141	1419
2	10.000	8,609	86.090
3	909	8,468	7697

Table 22: Comparing computation requirement on three different protocols

According to table 22, it is easy to see that algorithm 12 require less computation than algorithm 2 and 3. In table 22 phase 1-2 and 4 is included in algorithm 12, since the master hash chain has to be signed, and to do a signature require $7,468$ ms, that is added to the phase 2 according to table 19. Since the traffic hash chain hash to be change 10.000 times, phase 4 is also included in the calculation on required computation.

This is a simple model to compare these three different protocols, since it only look at a static condition and only one node, the initiator. Where the initiator broadcast messages to the neighbourhood, and the data packet is close to the maximum size for a longer time. With a master hash chain that contain $1.000.000$ master key and key up-date period of 1 second, the lifetime to the same master hash chain is nearly $11,5$ days. To include the responder there is necessary to do simulation, on a realistic scenario, which is not part of this master thesis.

6.4.5 Conclusion

This protocol is more secure against wormhole, DoS and insider attack than protocol in [31, 42, 23]. Since an initiator has to prove that he is intended to complete the authentication process, and the protocol also include end-to-end authentication. The protocol includes an end-to-end authentication protocol; where destination node is able to detect the manipulated message, which may be manipulated in a wormhole attack or by a compromised node. There is also strong indication on that the protocol is faster, and require less energy than protocols according to [31, 42, 23], since it only require public and private key operation the first time a node is joining the network, and when a node have to change master the hash chain. But the protocol requires that all legitimate nodes have synchronised clock, with a deviation much less than 1 second.

6.5 Recommendation

The new authentication protocol is based on the use of public/private key operation, hash function and HMAC function. In following section there is some recommendation on crypto algorithm, hash/MAC function, key size and key up-date period.

6.5.1 Public/private encryption algorithm

The protocol does not require a new signature in a protocol execution. The signature and generation to the master hash chain can be done before a node joining the network, or before the master hash chain has to be authenticated. But the verification of the signature and certificate is essential in the protocol execution, and should not require heavy computation. Since nodes that responds on an authentication request has to verify the certificate and signature to the initiator, when the initiator is a new node or the initiator has changes master hash chain.

Each node has to carry out signature of the master hash chain (for changing mater hash chain). If a node is authenticated with n node, it has to do n verification of certificate and signature. Since this protocol require more signature verification then signature generation, it is required that the verification is fast.

According to table 1, 2 (in section 4.7.4) and table 6 (in section 6.4) RSA has the property that the verification of a signature is fastest comparing to ECDSA and DSA, and the signature generation is slow comparing to ECDSA and RSA. Hence, RSA is a preferred algorithm, which is also according to the consideration in section 4.7.4 and 6.4.

6.5.2 hash function

The hash function has a main function in this protocol, both in generation and verification of hash chain key, and must be fast. In addition, it must be secure and hard to break (to invert and find collision) since it replace the public/private key operation and is fundamental for trusting the authentication. In table 3 MD-5 is faster than SHA-1, but

SHA-1 is preferred as it is more secure then MD-5. It is also known that SHA-1 has a security fault of 80 bits to resist the birthday paradox.

6.5.3 Message Authentication Code (MAC)

The protocol is based on authenticated message at each hop between the source and the destination node. This requires a fast MAC function, otherwise this will cause more delay. According to table 3 in section 4.7.4, we see that HMAC/MD-5 is approximately three times faster than AES encryption, and is preferred when only consider the speed. But HMAC/SHA-1 is already preferred in the hash chain and is more secure; to reduce the complexity and program size the HMAC/SHA-1 is preferred.

6.5.4 The key size

In table 36 and 37 in the appendix, the recommended key size from NIST is present. If we assume that the life time of a node is smaller than 10 years. Then the choice is balance between security and the efficient of the protocol. It seems that RSA with 1024 bit and SHA-1 with 160 bit are appropriate.

6.5.5 key up-date period

The key up-date period in this thesis is chosen to be 1 second. The choice is based on the practical consideration: keep the master hash chain as small as possible, to keep the time interval between authentication of traffic key as small as possible, to be able to have clock synchronisation even when some node for a time period does not have connection with necessary number of GPS satellite, and that a node that has a speed lower than 360 km/h should be able to receive at least two key up-dates (if each node have a radio coverage of at least 100 meter).

7 Features

The proposed protocol is appropriate to be used in node with different computation power and memory. Where number of public/private key operation and how large the hash chain has to be and fitted to the device computation power and memory space. It is also possible that a node is deployed with a mater hash chain that is according to the life time to the node. But to be efficient it requires that the hash chain key is stored.

If there is a network where it is not any need to control who is the user of the network it is possible to only use the end-to-end authentication protocol. The other case is if the treat from a wormhole attack, compromised node is small and every node trusts each other it is possible to only use the hop-by-hop authentication protocol.

This protocol, may be used in many situations e.g. to authenticate all device that is connected to a laptop with a wireless connection as a PDA, Cellular phone, etc. where the laptop is connected to another wireless network. In this case PDA and laptop represent a hop, and the end-to-end authentication is used between the PDA and the destination node.

8 Conclusion

In this master thesis a new authentication protocol is proposed, which consists of two parts; a hop-by-hop authentication protocol and an end-to-end authentication protocol. The hop-to-hop authentication protocol ensures that only legal node is part of the network, and that the message is authenticated on each hop between the source and destination node. But since this protocol is vulnerable to wormhole and insider attack, it is required to have an end-to-end authentication between source and destination node. These two protocols are based on the same crypto function and messages. But is different in that way, the hop-by-hop authentication protocol discloses authentication key immediately after a message is transmitted. This approach may not be used in the end-to-end authentication protocol, since it is not secure. In the end-to-end authentication protocol, the disclosure of authentication key is controlled, and the source node requires a responds from the destination node before the source node disclose the authentication key. In both part of the new authentication protocol the master hash chain replace signature generation and verification based on private/public key, which is time consuming to compute. But the first master hash chain, have a signature that has to be authenticated before the master hash chain is used to authenticate further protocol execution that include traffic and session key (that is based on hash chain). The new authentication protocol is not depended on which public/private key algorithm that is used (e.g. RSA, DSA or ECDSA). The choice on what algorithm that is preferred depends on the characteristic of the node which is used. If the ration between α (the ration between transmitting and computation power) and data rate is large according to 6.4.3, then RSA is preferred, if not ECDSA is preferred. Based on the test program in chapter D in appendix and consideration in section 6.4.4, there is strong indication on that the new authentication protocol, require less energy to complete authentication, compared to earlier proposed authentication protocol in [23, 42]. In addition the new authentication protocol is more secure against DoS attack, and the destination node is able to detect tampered (changed or manipulated) messages. A tampered message may be caused by an insider or wormhole attack. The earlier proposed protocol in [23, 42] is not secure against these attacks, and the destination node is not able to detect that a message is manipulated by an insider or wormhole attack. To protect against wormhole attack, is not easy, since it require tight clock synchronisation (much less then 10^{-6} second) between node, or public/private key operation on each message execution (that require lot of computation). The approach which is used in the new protocol is to be able to detect that a message is tampered between source and destination node.
The new authentication protocol is also flexible, where the size of all three hash chain may be fitted to different implementation, depended on the memory size, computation power, the lifetime to a node, since MANET may include different type of node with different computational power etc. There may also possible to have a chain of certificate, if more then one TTP is used to establish new certificate, but these certificate need cross certified or be signed by a root-CA. In this way there may be more than one rescue or-

ganisation that may issue certificate, but they have to be certified to issue certificate from one organisation.

Bibliography

[1] N. Aboudagga, M. T. Refaei, M. Eltoweissy, L. 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, New York, NY, USA, 2005. ACM Press.

[2] N. Aboudagga, M. T. Refaei, M. Eltoweissy, L. 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, New York, NY, USA, 2005. ACM Press.

[3] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. *Proceedings of Network and Distributed System Security Symposium 2002 (NDSS '02)*, 2002.

[4] I F Blake, G Seroussi, and N P Smart. *Advance in Elliptic Curve Cryptograpgy, London Mathematical Society Lecture Note Series 317*. Cambridge University Press, 2005.

[5] M. Brown, D. Cheung, D. Hankerson, J. L. Hernandez, M. Kirkup, and A. Menezes. Pgp in constrained wireless devices. *In 9th USENIX Security Symposium, August 2000*, 2000.

[6] S. Capkun, L. Buttyán, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transaction on Mobile Computing, Vol. 2, No. 1, 2003, pp. 52-64*, 2003.

[7] W. Du, R. Wang, and P. Ning. An efficient scheme for authenticating public keys in sensor networks. In *MobiHoc '05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 58–67, New York, NY, USA, 2005. ACM Press.

[8] L. Ertaul and N. Chavan. Security of ad hoc networks and threshold cryptography. *Wireless Networks, Communications and Mobile Computing, 2005 International Conference on Volume 1, 13-16 June 2005 Page(s):69 - 74 vol.1, Digital Object Identifier 10.1109/WIRLES.2005.1549386*, 2005.

[9] Internet Engineering Task Force. Mobile ad-hoc networks working group. http://www.ietf.org/html.charters/manet-charter.html. (Visited Dec. 2005).

[10] Internet Engineering Task Force. RFC 2202 test cases for HMAC-MD5 and HMAC-SHA-1. http://www.ietf.org/, 1997. (Visited Oct. 2006).

[11] Internet Engineering Task Force. RFC 2501, mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. http://www.ietf.org/, 1999. (Visited Dec. 2005).

[12] Internet Engineering Task Force. RFC 3561, ad hoc on-demand distance vector (AODV) routing. http://www.ietf.org/, 2003. (Visited Dec. 2005).

[13] Internet Engineering Task Force. RFC 3626, optimized link state routing protocol. http://www.ietf.org/, 2003. (Visited Dec. 2005).

[14] Internet Engineering Task Force. RFC 3684, topology dissemination based on reverse-path forwarding (TBRPF). http://www.ietf.org/, 2004. (Visited Dec. 2005).

[15] D. Halliday and R. Resnick. *Fundamentals of physics, 3ed*. John Wiley and Sons, 1988.

[16] A. Hodjat and I. Verbauwhede. Energy cost of secrets in ad-hoc networks. *In IEEE Circuits and Systems workshop on wireless communications and networking, Sept. 2002*, 2002.

[17] K. Hoeper and G. Gong. Models of authentication in ad hoc networks and their related, network properties. *University of Waterloo*, 2005.

[18] Y. Hu. and A. Perrig. A survey of secure wireless ad hoc routing. *Security & Privacy Magazine, IEEE Volume 02, Issue 3, May-June 2004 Page(s):28 - 39, Digital Object Identifier 10.1109/MSP.2004.1*, 2004.

[19] Y.-C. Hu, A. Perrig, D. B. Johnson, and P. Leashes. A defense against wormhole attacks in wireless networks. *0-7803-7753-2/03/ 2003 IEEE, pp. 1976-1986*, 2003.

[20] ITU International Telecommunications Union. The directory: Public-key and attribute certificate frameworks. *ITU-T Recommendation X.509, Information technology - Open Systems Interconnection, March 2000*, 2000.

[21] L. Lamport. Password authentication with insecure communication. *Commun. ACM*, 24(11):770–772, 1981.

[22] Shamus Software Ltd. Multiprecision Integer and Rational Arithmetic C/C++ Library, MIRACL,version 5.10. http://indigo.ie/ mscott/, 2006. (Visited Aug. 2006).

[23] B. Lu and U. W. Pooch. A lightweight authentication protocol for mobile ad hoc networks. *International Journal of Information Technology Vol. 11 No. 2, pp.119-135.*, 2005.

[24] H. Luo, P. Zerfos, S. Lu J. Kong, and L. Zhang. Self-securing ad hocwireless networks. *Proceedings of the Seventh International Symposium on Computers and Communications (ISCC'02)*, 2002.

[25] A. Menezes, P. Van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC press, 1996.

[26] Microsoft. Microsoft visual C++ 2005 version 8.0.5072.42. http://msdn2.microsoft.com/en-us/visualc/default.aspx, 2006. (Visited Oct. 2006).

[27] National Institute of Standard and Technology. Special publication 800-57, recommendation for key management-part 1. *Federal Information Standards, National Bureau of Standards, U.S. Department of Commerce, August, 2005*, 2005.

[28] National Institute of Standards and Technology. Mobile ad hoc networks. http://www.antd.nist.gov/. (Visited Dec. 2005).

[29] U.S. DEPARTMENT OF COMMERCE/National Institute of Standards and Technology. FIPS PUB 186-2,digital signature standard (DSS). *FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION,January, 2000*, 2000.

[30] U.S. Department of Transportation and U.S. Department of Defense. Federal Radionavigation Plan.

[31] A. Perrig, R. Canetti, D. Tygar, and D. Song. The TESLA broadcast authentication protocol. *Technical Report 2, RSA Laboratories*, 2002.

[32] A. Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.

[33] W. Stallings. *Network Security Essential Sec. ed.* Prentice Hall, 2003.

[34] D. R. Stinson. *Cryptography Theory and Practice*. Chapman and Hall/CRC, 2002.

[35] T. Suen and A. Yasinsac. Ad hoc network security: peer identification and authentication using signal properties. In *Systems, Man and Cybernetics (SMC) Information Assurance Workshop, 2005.Proceedings from the Sixth Annual IEEE 15-17 June 2005 Page(s):432 - 433*. IEEE, 2005.

[36] SiRF Technology Inc. SiRFstarIII GPS Single chip solution with high performance GPS-receiver in a small form factor. http://www.sirf.com/, 2006. (Visited Aug. 2006).

[37] W.Dai. Crypto++ 5.2.1 Benchmarks on most commonly used cryptographic algorithms on a Pentium 4 2.1 GHz processor under Windows XP SP 1 and Opteron 1.6Ghz processor under Linux 2.4.21. http://www.eskimo.com/ weidai/benchmarks.html, 2006. (Visited Aug. 2006).

[38] A. Weimerskirch. Authentication in ad-hoc and sensor networks. *Doktor Degree Thesis at Ruhr-University Bochum*, 2004.

[39] E. WINJUM, P. SPILLING, and Ø. KURE. Ad hoc networks used in emergency networks: The trust metric routing approach. *Repport from Norwegian Defence Research Establishment,FFI/RAPPORT-2005/04015*, 2005.

[40] J. Yao and G. Zeng. Key agreement and identity authentication protocols for ad hoc networks. *Proceedings of International Conference on Information Technology: Coding and Computing (ITCC'04)*, 2004.

[41] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network Journal Vol. 13 No. 6,1999, pp.24-30.*, 1999.

[42] S. Zhu, S. Xu, S. Setia, and S. Jajodia. LHAP: A lightweight hop-by-hop authentication protocol for ad-hoc networks. In *ICDCSW '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 749, Washington, DC, USA, 2003. IEEE Computer Society.