Ensuring Security in Ad Hoc Networks

Jimmy Kurian
Helsinki University of Technology
Telecommunications Software and Multimedia Laboratory
Jimmy.Kurian@hut.fi

Abstract

Security is of major concern in Ad Hoc networks. Achieving security in ad hoc networks is challenging due to vulnerable wireless link, hostile environment, cooperative algorithms, lack of a clear line of defense and dynamic topology. In an ad hoc network, the security requirements for different services range from highly security-sensitive military tactical operations to instantaneous classroom applications. This paper proposes a security architecture for service access in ad hoc networks.

KEYWORDS: Ad Hoc Networks, Security, Public Key Cryptography

1 Introduction

An ad hoc wireless network is a collection of two or more devices equipped with wireless communications and networking capability. Ad hoc devices are capable of detecting the presence of other such devices and perform necessary handshaking to allow communications and the sharing of information and services. Devices that are within each other's radio range communicate directly via wireless links, while those devices which are far apart rely on intermediate devices to relay or forward the messages from the source towards the destination. An ad hoc wireless network does not rely on any fixed network infrastructure and is self-organizing and adaptive. This means that a network could be formed or a formed network could be de-formed dynamically without the need for any system administration. The devices in the network are free to move arbitrarily, joining or leaving the network at any point of time, resulting in a highly dynamic network topology.[5]

The ability of an ad hoc mobile device to act as a server, providing a service, will depend on its computational power, memory, storage and battery capacity. Devices in an ad hoc network can be of different types such as palmtop, laptop, mobile phone etc. The heterogeneity of devices implies that some devices are more powerful than others, and some can act as servers providing some service while others can only be clients.[5]

Security is an important issue in ad hoc networks. Security is difficult to achieve, mainly because of the vulnerability of the wireless link, the limited physical protection, the sporadic nature of connectivity, the dynamic topology and membership, the absence of a certification authority, and the lack of a centralized monitoring authority [6].

The security requirements for different applications or services will vary in an ad hoc network. It also depends upon for what purpose the application is used. Applications in an ad hoc network range from military tactical operations to civil rapid deployment such as emergency search-and-rescue missions, data collection/sensor networks and instantaneous classroom/meeting room applications [4].

This paper proposes a security architecture for service access in ad hoc networks. It introduces an algorithm for handling service access. We will not address the issues related to routing. Security in routing is another area which is intensively researched and many solutions have been proposed [2, 3]. We will be concentrating on the issues related to service access in ad hoc networks.

The rest of the paper is organized in the following manner. In Section 2, we will discuss about the various security issues due to the nature of wireless ad hoc networks. It also sets the security goals for any security solution. In Section 3, we will discuss about the proposed solution along with the security architecture for ensuring security in ad hoc networks. In Section 4, we will check whether we achieved the security goals discussed in Section 2. We will conclude the paper in Section 5.

2 Security Challenges

The nature of wireless ad hoc networks makes them vulnerable to attacks. Achieving security is challenging due to the following features of ad hoc networks [7, 4];

Vulnerable wireless Link
The use of wireless link makes ad hoc networks susceptible to passive and active attacks. Unlike wired networks, which have several lines of defense like firewalls and gateways, attacks on a wireless ad hoc network can come from any direction and can target any device. Passive attacks like eavesdropping violates confidentiality, giving access to secret information. Active attacks ranges from message modification and replay to impersonation, which violates authentication, integrity, availability and non-repudiation. This means that every device should be prepared for protecting itself from any attacks.

Hostile environment
The devices in an Ad hoc network are autonomous units, which roam around in a hostile environment. Devices lack physical protection and are always under the threat of being captured and compromised. Further, a device is always

under the threat of being attacked by other malicious or compromised devices in the network. This means that every device should be prepared to operate in a mode that trusts no peer. Security solutions in a cooperative manner are always under risk. Adversaries can exploit the situation that lacks a centralized authority and can design attacks which breaks the cooperative algorithms. Every device should be capable of making its own security decisions without trusting other peer devices.

Dynamic topology and membership
An Ad Hoc network is very dynamic because of frequent changes in both its topology and membership. Devices join and leave the network at any time. Due to this, selecting certain devices to become administrative authority does not always work because they can always leave the network at any time. Further, there can be situations where compromised devices are elected for becoming the administrative authority or the administrative devices becoming compromised at a later stage.

Scalability
The number of devices in an Ad hoc network may range from two to even thousands of devices. So the security mechanism should be scalable enough to handle this situation.

2.1 Security Goals

The following security requirements are the goals to be achieved by any security solution in ad hoc networks[7];

- **Authentication** : Ensures the identity of the device with which the communication is done. This avoids impersonation.
- **Availability** : Ensures that the eligible devices are able to get the required services despite denial of service attacks.
- **Confidentiality** : Ensures that secret information or data is never disclosed to unauthorized devices.
- **Integrity** : Ensures that a message received is not corrupted.
- **Non-repudiation** : Ensures that a device cannot deny a particular action done by it at a later stage. This could help for the detection of compromised devices.

3 Ensuring Security

Any security solution that demands a cooperative approach is vulnerable to attacks. The solution proposed in this paper suits for a hostile environment, where no devices trust any other devices for making a decision. The paper assumes that each device has the necessary resources for Public-Key Cryptography. We won't be addressing the issues regarding the physical security of devices. Ideally, cryptographic information (mainly the private key) would be kept safely in a tamper resistant card.

3.1 The Environment

Each device possess a public/private key pair which is denoted as (K_{pub},K_{pri}) where K_{pub} is the public key and K_{pri} is the private key. Since a key is unique, K_{pub} is unique and thus H(K_{pub}), the fingerprint of K_{pub}, would also be unique. H(K_{pub}) of a device acts as its identifier in an Ad Hoc network. The device may also have certificates obtained from CAs. This is required if they need to access or provide some services to other unknown devices.

Every device sends its H(K_{pub}), along with a small description, to other devices in the network. Further, upon request, each device should be able to send its K_{pub} and certificates to other devices. Similarly, a device should also be able to request and receive K_{pub} and certificates from other devices.

3.2 Device Classification

A device D identifies all other devices in the network into three categories.

1. **Trusted Devices** : These are the devices which are trusted by the device D. The information about these devices are stored in the Trusted DB (explained below in Section 3.3) of D.
2. **Certificate-Trusted Devices** : These are the devices which posses a certificate issued by a trusted CA of the device D. These devices have come into contact with D earlier and D has successfully verified their certificate. The information about these devices are stored in the Trusted DB of D.
3. **Un-trusted Devices** : These are other devices in the network which does not fall into the above two categories of devices. This category may also contain devices which could be certificate-trusted by D. But since, there was no need to verify their certificates, they are still in the group of un-trusted devices.

3.3 The Security Architecture

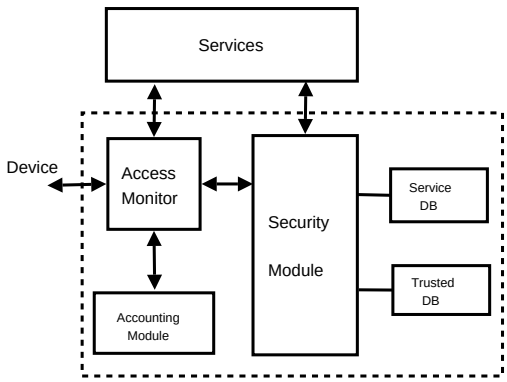

Figure 1: The Security Architecture

This section presents the Security Architecture (Figure 1) proposed by this paper. Each device in the ad hoc network

has the architecture built into them. The device acts as a server giving access to its own services or acts as a client while trying to access services from other devices. The scope of the access obtained is limited to that particular device only. A device itself makes the access decisions for its services rather than in a cooperative manner where a group of devices make the access decision. Devices operate in a mode that trusts no peer for making decisions and is based on the fact that every device in an ad hoc network is completely vulnerable to attacks. The security architecture makes it possible to grant access for trusted and certificate-trusted devices to specific services at the same time restricting access to un-trusted devices. However, if required, un-trusted devices may be authorized based on some type of user interaction and thus access may be granted.

This architecture authenticates only devices and not users. It means that if a device is stolen or borrowed, it can be used as if it was used by its real owner. If user authentication is needed, supplementary security methods such as the entry of a username and password, must be used.

Trust Level of Devices : It is possible to have different levels of trust for the devices, which determines the level of access to services. A device can fall in one or more levels of trust. A device can be given specific access for certain services also.

Security Levels of Services : It is possible to have different levels of security for the services, which determines the required trust level of the devices for accessing it. There can be certain services which does not need any authorization for access, thus open to all the devices.

3.3.1 Architecture Components

1. **Access Monitor** : Access monitor handles the connection requests. It stores information on existing sessions and upon a connection request checks whether the session is already authenticated and authorized. Otherwise it requests access from the Security Module. The access monitor handles the Accounting Module.

2. **Accounting Module** : The Accounting Module stores various accounting information of different events which occur during the access of the services. This module could support non-repudiation.

3. **Security Module** : The Security Module is responsible for managing information in Trusted DB and Service DB. It also handles the policy rules for making authorization decisions on available services. The Security Module grants access to services based on the trust level of the requesting device and the security level of the service(s) requested. These information are taken from the Trusted DB and the Service DB.

4. **Service DB** : This database keeps the information regarding various services run by the device. It also stores information regarding the security level of each service, which determines the required trust level of the devices for accessing it. It also keeps information like for accessing the service whether encryption is required for ensuring confidentiality of data. A service should be registered in the Service DB for making it accessible by other devices.

5. **Trusted DB** : This database keeps the information regarding;

   - Trusted devices : the database keeps information such as H(K_{pub}), K_{pub} and Display-Name for each device. It also keeps information regarding the trust level(s) of each device.
   - Certificate-Trusted devices : the database keeps information such as H(K_{pub}), K_{pub}, Display-Name and certificate details of each device.
   - Revoked Certificate-Trusted devices : the database keeps information regarding the revoked certificates. This helps in restricting future access with the same certificate.
   - Trusted CAs : the database keeps information such as K_{pub} and Display name of each CA.

3.4 Device Identification

When a device enters into the network, it starts receiving H(K_{pub}) from every other device in the network. It compares each H(K_{pub}) it received with the Trusted DB. The device distinguishes each of those devices as trusted, certificate-trusted and un-trusted devices. But, this doesn't mean that, those devices are really whom they claim to be. There can be some devices which sends false H(K_{pub}), thus impersonating some other devices. Authentication will be done before the access of services or in case of conflicts.

Every un-trusted device will be shown according to the description they send along with their H(K_{pub}). But each trusted and certificate-trusted device will be shown according to the Display-Name, which is stored in the Trusted DB.

3.5 Device Registration

Trusted Device

Consider two devices A and B. B receives the fingerprint of A's public key through some other secure means (by phone or by meeting A's owner in person). Now B wants to add A in its list of trusted devices. B selects the device from the network whose H(K_{pub}) matches with the fingerprint which it received. Now B requests the device, which it selected, for its K_{pub}. When it receives the K_{pub}, B computes the fingerprint of that K_{pub} and checks it to confirm. This is required to make sure that the device which it selected is having the same fingerprint or not. At this stage, B has the K_{pub} of the other device and authenticates it with a challenge-response. Now B is sure that the other device is A itself and it registers A in the Trusted DB as a trusted device. It stores information of A such as H(K_{pub}), K_{pub} and Display-Name in its Trusted DB. As the Display-Name, the user of B can select the description which A gives or the user can enter a new value for it. When registered, the B's owner can set the trust level(s) of A such that it can access a specific service or a group of services. A can be given specific authorizations for certain services also.

Certificate-Trusted Device
Consider two devices A and B. A wants to access some service in B, but A is not a trusted device of B. B checks whether A has a certificate which is certified by a trusted CA according to B's Trusted DB. For this a certificate chain verification may be necessary to make it sure that the given certificate is well-formed, valid, properly signed, and trustworthy. Since a central repository of certificates is not available, a device must also carry the certificates leading to the root CA. After verifying the certificate, B is sure that the other device is A itself and it registers A in the Trusted DB as a certificate-trusted device. It stores information of A such as H(K_{pub}), K_{pub}, Display-Name and certificate details in its Trusted DB. As the Display-Name, the user of the B device can select the description which A gives or the user can enter a new value for it. When registered, A can be given the default rights according to the certificate or B's owner can set the trust level(s) of A such that it can access a specific service or a group of services. A can be given specific authorizations for certain services also.

3.6 Device Revocation

Under various circumstances a device must be revoked. Such circumstances occur for example, when a private key is compromised or when a device is stolen, This section deals how a device is revoked.

Trusted Device
In the case of a trusted device, the information regarding the revocation is obtained by the user from some other secure means. The user updates the trusted DB thus revoking the device from further access. In the case of trusted devices, the user just have to delete the revoked device data from its trusted DB.

Certificate-Trusted Devices
A certificate may become invalid prior to the expiration of the validity period. Certificate revocation is done in conventional networks with the help of Certificate Revocation Lists(CRLs)[8]. CRLs contain information about revoked certificates and are published by Certification Authorities (CAs). But it requires external network connectivity (for example, internet connectivity[1]) for ad hoc devices to connect to the CAs or to the central repositories from where CRLs could be retrieved. It could be thought that some of the devices capable of ad hoc networking will have the ability for external network connectivity also, but they choose to stay away because they need to pay for it. If external network connectivity could be achieved, then CRLs could be retrieved periodically. In lieu of or as a supplement to checking against a periodic CRL, it may be necessary in security-sensitive services, to obtain timely information regarding the revocation status of a certificate. This could be obtained with the help of Online Certificate Status Protocol(OCSP)[9]. But this also requires network connectivity, as discussed above, to connect to the centralized servers running the certificate validation protocol. However the usage depends upon how security-sensitive the requested service is and according to the need.

If no external connectivity could be achieved, then certificate revocation presents a major challenge in ad hoc environments. A lot of research is going on in this area.

Unlike revoked trusted devices, whose data is deleted, the revoked certificate-trusted device's data should be stored in the trusted DB. This helps in restricting the access to the services with the same certificate in the future.

3.7 Access Of Services

When a service access request from a device comes to the Access Monitor it checks whether the session is validated or not. If the session is not validated, then it passes the access request to the Security Module. The Security Module checks the trusted DB to identify the category of the device.

Trusted Device and Certificate-Trusted Devices
The Security Module performs authentication of the device. If the device could be authenticated, it performs authorization and grants access if the device is eligible for accessing the service. In the case of Certificate-Trusted Devices, if required, Security Module might check for certificate revocation.

Un-trusted devices with a valid certificate
If the request for access is from a un-trusted device, check is done to verify whether it has any valid certificate which is certified by a trusted CA. If it has a valid certificate, then the device could be registered as a Certificate-Trusted device and access could be granted if it is eligible for accessing the service.

Un-trusted devices without a valid certificate
These un-trusted devices which don't have a valid certificate may be allowed access by the user of the device which runs the service. The user can set the "manual permission option" ON, which asks for user interaction when a un-trusted device tries to access a service. Upon giving the manual permission, the user should also set the trust level of the device or give specific authorization permission for the requested service. Authentication is required to make it sure that the device is having the public key which it claims to have. Manual permission is valid only for that particular session and does not grant future access automatically.

After the authentication and authorization of the devices, a valid session is created. The Access Monitor requests the Accounting Module for starting the accounting process. Mutual authentication between the devices is done to avoid impersonation. But authentication is done only when some access or communication between the devices is required. This helps to reduce the work load on devices required for authentication.

3.7.1 Access Algorithm

The algorithm shows the steps taken by a device while handling a service access request.

The following are the functions used in the algorithm.

- authentication() : returns true if authentication for the device was successful. Else it returns false. Authentication of the device is always necessary. In the case of request coming from a Trusted Device or Certificate-Trusted Device, the K_{pub} is obtained by the Security Module from the Trusted DB. Otherwise it is obtained from the un-trusted device itself. Authentication is done with the help of a challenge-response.
- authorization() : returns true if authorization for the device was successful or if authorization is not required for the requested service. Else it returns false. Authorization might be required to make sure that the authenticated device can be granted access to the requested service. The Security Module checks the Trusted DB and the Service DB to determine whether the device is eligible for the requested service access. Some services might not require authorization for access and can be accessed by any authenticated devices.
- trustedDevice() : returns true if the device is a Trusted device according to its Trusted DB. Else it returns false.
- certificateTrustedDevice() : returns true if the device is a Certificate-Trusted device according to its Trusted DB. Else it returns false.
- checkValidCertificate() : returns true if the device could provide a valid certificate. Else it returns false.
- registerCertificateTrustedDevice() : returns true if the device could be registered as a Certificate-Trusted device.
- manualPermissionON : boolean value indicating whether user has set the manual permission option.
- getManualPermission() : returns true if the user has given permission manually for the device which is accessing the service. Else it returns false.
- getTrustLevel() : returns true if the user sets the trust level of the device or grants specific authorization permission for the requested service. Else it returns false.
- grantAccess() : grants the access to the service
- denyAccess() : denies the access request to the service.

The algorithm is given below.

```
// gets the access request for the service
getAccessRequest();

if ( trustedDevice())
{
   if ( authentication() && authorization() )
   {
      // Authentication and Authorization passed
      grantAccess();
   }
   else
   {
      // Authentication and/or Authorization failed
      denyAccess();
   }
}
else if ( certificateTrustedDevice() )
{
   // Check for certificate revocation if needed.
   if (authentication() && authorization())
   {
     // Authentication and Authorization passed
     grantAccess();
   }
   else
   {
     // Authentication and/or Authorization failed
     denyAccess();
   }
}
else
{
   // un-trusted device

   if ( checkValidCertificate() )
   {
      // device has a valid certificate
      if ( registerCertificateTrustedDevice() )
      {
         // device registered as certificate-trusted
         if (authentication() && authorization())
         {
           // Authentication and Authorization passed
           grantAccess();
         }
         else
         {
           // Authentication and/or Authorization failed
           denyAccess();
         }
      }
   }

   if ( manualPermissionON )
   {
      // device does not have a valid certificate
      // or it failed to register as certificate-trusted

      // User has set the manual permission option
      if ( getManualPermission() && getTrustLevel())
      {
        // user gave the manual permission
        // and selected the Trust Level of the device

        // required to avoid impersonation
        if (authentication())
        {
          grantAccess();
        }
        else
        {
          denyAccess();
        }
      }
      else
      {
        denyAccess();
      }
   }
   else
   {
     denyAccess();
   }
}
```

3.8 Transparency To End-Users

The proposed solution is very transparent to the end users and demands no technical expertise from them. The places where the user interaction is needed is very limited such as trusted and certificate-trusted device registration. We will consider a situation with two end users, Alice and Bob.

Bob's device D_b wants to add Alice's device D_a to its list of trusted devices. Alice gives Bob the finger print of D_a's public key through some other secure means (by phone or by meeting him in person). When Bob comes to the ad hoc network, all the devices will send their H(K_{pub}) along with a description to D_b. Bob selects the device which is having the H(K_{pub}) which is identical to the fingerprint received from Alice (a device which claims to be D_a). Now

D_b requests D_a for its Public Key, K_{pub}. It then authenticates D_a with a challenge-response mechanism. This should be done to avoid impersonation. Now that D_a is authenticated, Bob stores the data of D_a in D_b as a trusted device. He might choose the same description, which is sent by D_a, as its Display name or enters some other text. Bob also sets the trust level(s) for D_a, which determines the level of access to services. Now D_a is a trusted device of D_b.

Now at some other point of time, when D_b and D_a comes in an Ad Hoc network, D_b is able to recognize D_a as a trusted device. Now when D_a tries to get access to D_b's services, D_b just have to verify that the device that claims to be D_a is D_a itself. This can be done by authentication. Note that, D_b does this only when an access comes from D_a.

Here the entire process is very transparent to Bob and Alice. For example, consider the first step where D_a should be added in D_b's trusted list. Bob sees a list of devices in the network according to their fingerprints (to him it is just a number). He selects the device which matches with the number given to him by Alice (actually D_a's finger print) and adds it to his list of trusted devices. He might add a description like "Alice PDA" for the device. Next time when D_a comes in the network, D_b shows "Alice PDA" in the list of D_b's trusted devices, instead of the fingerprint, and Bob recognizes the device according to his own description he entered. When D_a tries to access D_b's services, D_b automatically authenticates D_a which can be entirely transparent to Bob.

3.9 Usage Scenario

This section explains a usage scenario where a user tries to use a service.

Bob works in a Company "XYZ". He wants to print a secret document. His device D_b needs to send the document to the printer device. The following scenarios occur according to the printer.

- Same Department's printer : Here Bob needs to print the document in his own department's printer. It is already there in D_b's list of trusted devices. D_b just has to authenticate the device (whether it is the same as it claims to be) and can send the file for printing. The printer does not know D_b. But D_b has a certificate which proves that it is a device of an employee of XYZ. Thus printer authorizes D_b for using the printing service.

- Some other Department's printer : Here Bob goes to some other department in his own company and wants to print the document. But the printer is not there in the list of trusted devices of D_b. But the printer has a certificate from the System Admin of the company showing that it is a legitimate printer. System Admin is there in the list of trusted CAs of D_b. Thus D_b verifies the certificate and the printer. Thus the printer is saved as a certificate-trusted device of D_b. Now D_b sends the file for printing. The printer on the other hand recognizes D_b according to the certificate. But since Bob is from a different department, D_b is not allowed for color printing service.

- An Internet Cafe's printer : Here Bob goes to an Internet Cafe and wants to print the document. The printer is not there in D_b's list of trusted devices nor the printer has any certificates which D_b trust. But Bob can see the printer and can obtain its finger print manually (he sees it written on the printer or the Cafe gives him). He selects the device from the list and D_b verifies the printer. Bob sends the file for printing. The printer on the other side, requires manual permission from the cafe to start printing. Once it gets the permission, it allows D_b to use the printing service. But this permission is valid for only this session, and needs to get manual permission again if D_b wants to use the printing service again at some other time.

4 Analysis

This section discusses whether this solution was able to achieve the security goals set in Section 2. The solution is a combination of public key cryptography along with the help of certificates. Each device possess a public/private key pair. Public keys are distributed to other devices and helps as the identification of the devices in the network. Private keys are kept confidential to individual devices.

Confidentiality and integrity of data is assured with the help of asymmetric keys. Encryption is possible and is used according to the security demands of the service. To improve efficiency, it is better to use asymmetric keys for the exchange of a symmetric key generated by one machine and encrypted with the public key of the other device. Further communication is done with the help of the symmetric key. Non-repudiation could be achieved with the help of cryptographic keys and accounting.

Before the access of services, mutual-authentication between the devices is done to avoid impersonation. Authentication is not done until a service is accessed or a need for communication arises. This helps in reducing the workload on the devices required for authentication. But if a device tries to impersonate some other device by sending H(K_{pub}) of that device to all others, and if it creates a conflict, authentication could be demanded.

This solution does not assure the availability of services to other devices, in case of a denial of service attack. And this solution does not discuss about the physical protection of the devices. For certificate revocation, this solution suggests contacting the CAs periodically, if the device is capable of external network connectivity. This is based on the fact that any device is open for attack in an ad hoc environment, and thus cannot trust other peer devices for making a decision on revocation of devices. This could lead to denial of service attacks. Further, it could be thought that most of the devices capable of ad hoc networking will have the ability for external network connectivity also, but it chooses to stay away because we need to pay for it.

This solution also provides the facility for a device to be in the ad hoc network without the use of certificates. These are the devices which choose to communicate only with their Trusted devices. When it comes to communicate with any Un-trusted device, the decision is left to the user of the device. These devices does not require to have the complexity

needed for handling the certificates or certificate revocation.

This solution is free from man-in-the-middle attack since, trusted devices are registered with the help of data obtained from secure means. The certificate-trusted devices are trusted with the help of the certificates. Further, every device is identified in the network, with its public key. Each device operates in a mode that trusts no peer device. Cooperative algorithms are not used for making an access decision. Thus the solution does not specify the minimum number of devices required for the network. Further, it can scale to any number of devices also. Each device decides on its own and the scope of the access provided is limited to the device itself. Thus compromised devices are not able to bring the whole network down. Further, even if the network has a majority of compromised devices, it doesn't affect an un-compromised device.

5 Conclusion

This paper proposes a security architecture for service access in ad hoc networks. It also proposes an access algorithm which explains the steps taken by a device while handling a service access request. The solution is a combination of public key cryptography along with the help of certificates. Each device in the network is uniquely identified with its public key. The solution protects against various issues of vulnerable wireless link like active and passive attacks. It is scalable and does not depend on other devices. It can be used in a very hostile environment. Dynamic topology and membership does not affect the solution since a device makes the access decision on its own and avoids the use of cooperative algorithms.

References

[1] Charles E. Perkins, Jari T. Malinen, Ryuji Wakikawa, Anders Nilsson and Antti J. Tuominen. Internet connectivity for mobile ad hoc networks. *Wirel. Commun. Mob. Comput. 2002.*

[2] Panagiotis Papadimitratos and Zygmunt J. Haas. Secure Routing for Mobile Ad hoc Networks. *In Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002)*, San Antonio, TX, January 2002.

[3] Kimaya Sanzgiri, Brian Neil Levine, Clay Shields, Elizabeth M. Belding-Royer and Bridget Dahill. A Secure Routing Protocol for Ad Hoc Networks. *In Proceedings of the 10 th IEEE International Conference on Network Protocols (ICNP 2002).*

[4] Yongguang Zhang and Wenke Lee. Intrusion Detection in Wireless Ad-Hoc Networks. *In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, August 2000, Boston, Massachussetts.

[5] C-K Toh. Ad Hoc Mobile Wireless Networks: Protocols and Systems. *ISBN 0-13-007817-4*, 2002.

[6] Jean-Pierre Hubaux, Levente Buttyan, and Srdan Capkun. The Quest for Security in Mobile Ad Hoc Networks. *In Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC 2001).*

[7] Lidong Zhou and Zygmunt J. Haas. Securing Ad Hoc Networks. *IEEE Network*, 13(6):24-30, November/December 1999.

[8] R. Housley, W. Polk, W. Ford and D. Solo. RFC 3280 Internet X.509 Public Key Infrastructure, Certificate and Certificate Revocation List (CRL) Profile. *April 2002.*

[9] M. Myers, R. Ankney, A. Malpani, S. Galperin and C. Adams RFC 2560 X.509 Internet Public Key Infrastructure, Online Certificate Status Protocol - OCSP. *June 1999.*