On Demand Public Key Management for Wireless Ad Hoc Networks

Xia Li[1], Steven Gordon[2] and Jill Slay[1]
School of Computer and Information Science[1]
Institute for Telecommunications Research[2]
University of South Australia
Mawson Lakes, Adelaide SA 5095, Australia
{xia.li, steven.gordon, jill.slay}@unisa.edu.au

***Abstract*—A wireless ad hoc network is an autonomous system that is made up of collaborative mobile nodes. Wireless ad hoc networks can be dynamically set up without relying on any pre-existing infrastructure or central administration. Implementing public key management is a challenging issue in wireless ad hoc networks due to its salient nature of the network. Without an online third party, the public key certificate distribution is vulnerable to man-in-the-middle attacks. In this paper, we present On-demand Public Key Management (OPKM), a novel public key management scheme for wireless ad hoc networks. OPKM makes use of broadcasting technology and digital signature mechanism to provide key management service on demand, while protecting the certificate distribution against man-in-the-middle attacks. OPKM can be fully organised by nodes themselves without the need of any online trust third party.**

I. INTRODUCTION

A wireless ad hoc network is an autonomous system that is made up of collaborative mobile nodes equipped with wireless transceivers. Each node is able to communicate with other nodes within its transmission range. For those nodes that are far apart from each other, the communication will rely on intermediate nodes to relay the messages. Wireless ad hoc networks are dynamic because the nodes may move randomly and join or leave the network any time at their will. As a result, the neighbourhood and trust relationship may also change accordingly. Wireless ad hoc networks can be dynamically constructed without relying on any pre-existing infrastructure and central administration.

With the proliferation of wireless technology, wireless ad hoc networking is becoming an attractive solution to the services that need flexible set-up, dynamic and low-cost wireless connectivity. The ever-growing demands also raise great concerns on the security of wireless ad hoc networks, especially for security sensitive applications. Recent research [1] [2] has indicated that wireless ad hoc networks are prone to various malicious attacks ranging from passive eavesdropping to active information modification. Since it is unrealistic to assume that every node would behave honestly, the deployment of security service becomes crucial for information protection in the open environment of wireless ad hoc networks.

Public key cryptography [3] has been widely recognized and accepted as an effective mechanism for providing fundamental security services including authentication, confidentiality, integrity and non-repudiation. The essential pre-requisite of this cryptography is that all participating entities must know each other's public key. Conventional public key management is implemented with public key infrastructure (PKI), in which a trusted third party holds the public key certificates of all participating entities and acts as an online Certificate Authority (CA) to provide a public key verification service.

However, without infrastructure support, it is particularly challenging to implement the public key management in wireless ad hoc networks.

Firstly, no third party can be expected in wireless ad hoc networks. A trusted third party is an essential component of the conventional PKI for verifying participants' public key certificates. But in wireless ad hoc networks, such a third party may not exist and the public key management has to be operated in a self-organised manner by nodes themselves.

Secondly, distributed cooperation for multi-hop communications makes the public key distribution in wireless ad hoc networks vulnerable to man-in-the-middle attacks. A man-in-the-middle attack is a type of attack where an intermediate node may maliciously modify the message between the source node and destination node without letting either node know the message has been attacked. Since trust may not exist among all nodes in the network, there is no guarantee that a public key certificate will be transferred correctly without implementing any security mechanism, especially when an intermediate node is compromised and behaves maliciously.

Thirdly, the dynamic topology and free membership features of wireless ad hoc networks would challenge any public key management framework that is based on static node arrangement. The node movement and membership variation may dynamically change both the network topology and trust relationships between nodes. Such features require the public key management framework to

be flexible and dynamic. In other words, the public key certificates should be able to be distributable on demand.

In this paper, we utilise the broadcasting property of radio communications and self-signed public key certificates to propose a flexible public key management scheme for wireless ad hoc networks, which is able to overcome the above challenges. We call this novel approach "On-demand Public Key Management" (OPKM) scheme, in which every node is able to hold dynamically two hops neighbours' self-signed public certificates and distribute the public key certificates through multi-hop communication on demand in a verifiable way. In our scheme, every intermediate node may check the 1- and 2-hop neighbours' digital signatures, which guarantee that no single node may modify the public key certificate information during the distribution process. Our proposed scheme can be operated in a fully self-organised manner without relying on any central administration or CA.

The main concepts of the key distribution scheme are presented in this paper. Informal analysis of the scheme's security, as well as discussion of limiting performance factors is given. As an initial proposal, we do not attempt to provide a formal specification and analysis, nor detailed performance analysis. This will be part of the next stages of the scheme's development.

The remainder of this paper is organised as follows. We review the main proposed approaches on public key management for wireless ad hoc networks in Section II. Motivated by the limitations in the related work, we design an on-demand public key management scheme, which is presented in Section III. We informally analyse the robustness of our approach against man-in-the-middle attacks in Section IV. Some performance concerns are discussed in Section V. Finally, we summarise our contribution and future work in Section VI.

II. Related work

With many potential applications, significant research effort has been directed towards security issues in wireless ad hoc networking. Many of these security discussions are based on asymmetric key cryptography and assume that the public keys have already been distributed and known among all participating nodes. However, the public key management for wireless ad hoc networks has only recently started to gain attention. The current proposed approaches in this area could be generally classified into two categories: the distributed approaches with threshold cryptography and PGP-like certificate chain approach.

A. Distributed Approaches with Threshold Cryptography

In [4], Zhou and Haas propose a key management approach with threshold cryptography [5] to distribute CA functionality across a number of nodes as servers. In this approach, the authors assume that at least n server nodes exist and know the public keys of all nodes in the networks. Each server node may generate a partial digital signature with its pre-distributed share of the network private key. To verify a public key certificate, a request node may work out the whole digital signature by getting any k-out-of-n partial digital signatures from the server nodes. Such a certificate service may tolerate up to k-1 server nodes to be compromised.

In [6], Yi and Kravets present a practical key management framework for wireless ad hoc networks called Mobile Certificate Authority (MOCA) framework, which is similar to [4]. This approach also assumes that before setting up the networks, there is central authority to distribute all public keys to n MOCA nodes. But the MOCA nodes are chosen based on node's computation capability and physical security.

In [7], Khalili, et al., introduce the idea of an ID-based private key generation using threshold cryptography. In this approach, each node uses its network identity (ID) as its public key. The corresponding private key of the node can be generated in threshold manner by contacting at least k nodes that formed the public key generation service (PKG). This approach avoids the need for users to generate their own public keys and to then distribute these keys throughout the network, but requires that a node have direct contacts with at least k PKG nodes for its private key generation. Meanwhile, this approach is suspected to be able to extend to large-scale wireless ad hoc networks, since the initial PKG nodes are defined at the beginning of the network.

In [8], Kong propose a security scheme by using threshold cryptography to distribute CA's functionality to each local neighbourhood. When a node requests a certification service, a local coalition of k secret shareholders is formed on the fly as a CA. This approach requires centralized management for network initialisation. Meanwhile, the secret share update mechanism requires the network to be synchronized, which is hard to achieve in wireless ad hoc networks.

The main limitations of these distributed approaches with threshold cryptography are:

1) At least k nodes have to be pre-allocated as server nodes to serve the public key management before the network is setup. This goes against the self-organising nature of wireless ad hoc networks.
2) A node that wants to get the public key management service has to directly contact at least k-out-of-n server nodes. Otherwise the service is susceptible to the man-in-the-middle attack, since there is no guarantee that the intermediate node will not modify the public key information.

B. PGP-like Certificate Chain Approach

In [9], Capkun, et al., presents a fully self-organised public key management scheme for wireless ad hoc networks. In this approach, each node maintains two certificate repositories—non-updated and updated

certificate repository, in which the valid certificates are maintained in a certificate chain. When two users want to verify each other's public key, they merge their local certificate repositories and try to find appropriate certificate chains within the merged certificate repository. The operation of this approach can be fully self-organised without the need of any pre-allocated server nodes.

However, this PGP-like certificate chain approach also has several limitations:

1) It doesn't define how to protect the verification process against the man-in-the-middle attack if an intermediate node maliciously modifies the certificate chain at the time when two remote nodes want to authenticate each other by merging their certificate repositories.
2) The movement of dynamic nodes or a large number of public key revocations may cause the graphic certificate chains in each local repository to frequently become obsolete as the certificate chains become invalid. Meanwhile, the reconstruction of these local repositories is expensive and complicated.

From the above review, we can see that each of the categorized approaches has its own drawbacks and limitations. Targeting these limitations, we propose a novel public key management scheme for wireless ad hoc networks. The details of this scheme are described in the following sections.

III. ON-DEMAND PUBLIC KEY MANAGEMENT

For deploying public key cryptography in wireless ad hoc networks, a proper key management scheme is a prerequisite to such deployment for underlaying security services. In order to adapt the dynamic feature of wireless ad hoc networks, we present a novel approach, called "On-demand Public Key Management" (OPKM), for the networks. Our OPKM can be fully performed by nodes themselves without the need of any trusted third party in the network, while providing dynamic public key management services against various man-in-the-middle attacks.

A. Assumptions

OPKM uses broadcast technology of wireless transmission to distribute public key certificate information among all nodes in a wireless ad hoc network and employs digital signature to verify the broadcast message against man-in-the-middle attacks.

To simplify our discussion, we assume that each node in the network is equipped with omni-directional antenna for network communications and all links between the nodes are bi-directional. In addition, all nodes are assumed to be able to implement the necessary digital signature computation including both encryption and decryption. Finally, every honest node is assumed to join the network with a unique network identity (ID).

B. Overview of OPKM

OPKM is a dynamic approach that is flexible to adapt the neighbourhood status changes among the nodes in wireless ad hoc networks.

Before explaining our scheme, we define the notation used. Two nodes are called *1-hop neighbours* if they are within each other's transmission range. Two nodes that are beyond one hop distance but have at least one common 1-hop neighbour are called *2-hop neighbours*. By *multi-hop communication* we refer to communication between nodes that are more than two hops away from each other.

In OPKM, every node that first joins the network or enters into a new neighbourhood performs a proactive process by broadcasting its public key certificate as a request to its 1-hop neighbours. With the designed neighbourhood certificate distribution mechanism, every node is able to dynamically obtain all up-to-date public key certificates from the neighbours within its two hops distance. For those nodes that are more than two hops away from each other, their multi-hop public key certificate distribution can be initialised on demand, whenever the two nodes want to exchange their public key certificates for initiating subsequent secure communications.

Each node in the network maintains two tables for storing other nodes' public key certificate information. One table is called *neighbourhood certificate table*, in which a node dynamically stores the public key certificates of the neighbours within its two hops distance. The other table is called *network certificate table*, in which a node stores all public key certificates of the available nodes it knows in the network.

C. Basic Operation

A secure public key management scheme generally involves four processes: key generation, distribution, verification and revocation. We describe our OPKM scheme in detail according to these four processes.

1) Generation of Public Key Certificate

In OPKM, every node i is responsible for creating a public/private key pair itself before joining the network. One key is kept secret as its private key (PRK_i). The other is prepared to be publicly available as its public key (PUK_i). For exchanging public key information with others, node i issues a self-signed public key certificate C_i, which contains node i's identity ID_i, the node i's public key PUK_i, validity period of the certificate ΔT_i and its digital signature D_i. Digital signature D_i can be obtained by node i to apply a common hash function $H()$ to the contents of C_i and encrypt the hash value with PRK_i. The public key certificate of node i can be generally described as:

$$C_i = <ID_i, PUK_i, \Delta T, D_i>$$

$D_i = E_{PRK_i}[H(ID_i, PUK_i, \Delta T)]$

Using the self-signed certificate in OPKM is to provide the security service of non-repudiation and integrity.

2) Distribution of Public Key Certificate

In OPKM, we propose two processes for a node to distribute its public key certificates to other nodes. One process is called *neighbourhood certificate distribution*. The other is called *multi-hop certificate distribution*. The neighbourhood certificate distribution is designed for a node in the network to dynamically exchange its public key certificate with its neighbours within two hops. Based on the neighbourhood certificate distribution, the multi-hop certificate distribution is designed for a node to issue its public key certificate to another node that is more then two hops away.

Neighbourhood certificate distribution is a proactive process for a node to perform when it joins the network or move to a new position with its neighbourhood status changed. Every node in the network may be aware of its neighbourhood status by periodically broadcasting a *hello* message to its 1-hop neighbours. A node i triggers its neighbourhood certificate distribution process when a new 1-hop neighbour of node i is acquired. The neighbourhood certificate distribution of node i is performed through the following three steps:

Step 1: Node i first broadcasts a *PKC_REQ* request message M_i to all its 1-hop neighbours $N_1(i)$, in which node i's public key certificate C_i is included. This step can be described as:

$$i \rightarrow N_1(i): \quad M_i = \{ PKC\_REQ, C_i \} \tag{1}$$

Step 2: Each node $j \in N_1(i)$ that receives the *PKC_REQ* request firstly validates the request message M_i by verifying node i's digital signature with node i's public key PUK_i in the message M_i. The validation is to ensure that the message M_i originated from node i and is transmitted correctly. Then every node $j \in N_1(i)$ updates its both neighbourhood certificate and network certificate tables and rebroadcasts a *PKC_REP* reply message M_j after a chosen random time t_j. The value of time t_j is defined by node j to avoid the broadcast storm problem. The message M_j includes not only node j's public key certificate C_j, but also the public key certificates $C_{N_1(j)}$ of all j's 1-hop neighbours $N_1(j)$.

$$j \rightarrow N_1(j): \quad M_j = \{ PKC\_REP, C_j, C_{N_1(j)} \} \tag{2}$$

Every node $m \in N_1(j)$ that receives the *PKC_REP* reply message M_j from node $j \in N_1(i)$ will not rebroadcast the message, but simply update its neighbourhood certificate table and network certificate table, in both of which node i's public key certificate is added if it is new.

Step 3: the initial node i updates its neighbourhood certificate table and network certificate table according to the information in the *PKC_REP* reply message M_j received from node $j \in N_1(i)$. After a defined time T_i, node i broadcasts a *PKC_UPDATE* update message M_i to its 1-hop neighbours $N_1(i)$. The time T_i is defined to ensure node i is able to receive all *PKC_REP* reply messages from its 1-hop neighbours $N_1(i)$. In the *PKC_UPDATE* update message M_i, it contains node i's public key certificate and all public key certificates $C_{N_1(i)}$ of node i's 1-hop neighbours $N_1(i)$.

$$i \rightarrow N_1(i): \quad M_i = \{ PKC\_UPDATE, C_i, C_{N_1(i)} \} \tag{3}$$

Every node that receives the *PKC_UPDATE* message M_i from the initial node i will update its neighbourhood certificate table and network certificate table accordingly if any new public certificate is contained in the message.

This *PKC_UPDATE* update message is necessary, especially for the case that two nodes are not within each other's two hops range before node i joins into their neighbourhood, but they become each other's 2-hop neighbour after that.

By taking the above three steps, every node in the network is able to dynamically obtain all public key certificates of its neighbours within two hops distance. An example of these three steps is illustrated in the Fig. 1.

Multi-hop certificate distribution is designed for the two nodes that are beyond each other's two hops communication range to exchange their public key certificate. Based on the neighbourhood certification distribution, OPKM multi-hop public key certificate distribution can be initialised on demand.

In OPKM, the multi-hop certificate distribution process complies with the principle that every intermediate node will not rebroadcast a message that has already been transferred by its two 1-hop neighbours. An intermediate node may confirm such information by checking the node list in the message with its neighbourhood table. If two nodes in the list are found to be 1-hop neighbours of next intermediate node, the next intermediate node will discard the message. This principle is designed for some security concerns, which will be discussed in Section IV.

Based on the defined principle, we describe our multi-hop certification distribution process in detail. Now we suppose a source node S wants to exchange its public key certificate with a destination node D that is more than two hops away from node S. The distribution process is presented as followings:

Node S initialises the distribution process by broadcasting a *PKC_DT* request message MD_s to its 1-hop neighbours $N_1(s)$. The *PKC_DT* request message MD_s contains node S's public key certificate and the intended destination node D's identity ID_d.

$$S \rightarrow N_1(s): \quad MD_s = \{PKC\_DT, ID_d, C_s\} \tag{4}$$

Each node $i \in N_1(s)$ validates the *PKC_DT* request message MD_s by verifying the digital signature D_s with the public key PUK_s stored in its neighbourhood certificate table. Then node i appends its public key

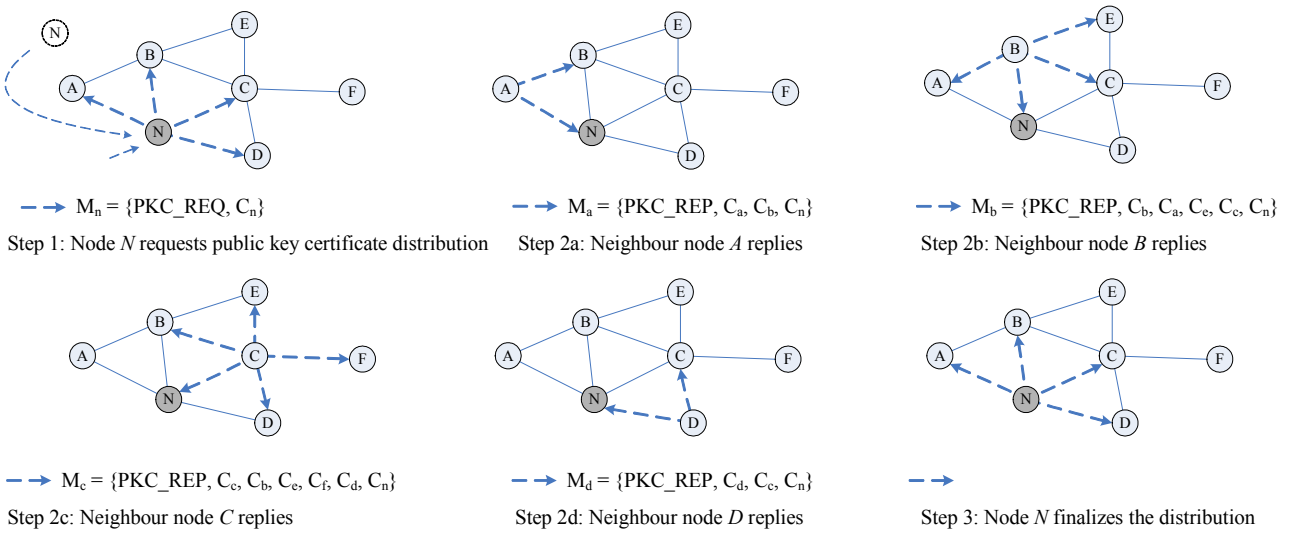

Fig. 1. Illustration example for neighbourhood public key certificate distribution

certificate C_i to the message MD_s and encrypts the whole contents with its private key PRK_i as its digital signature. Now the message is organised as a new message MD_i. Node i broadcasts the message MD_i to its 1-hop neighbours $N_1(i)$.

$$i \rightarrow N_1(i): \quad MD_i = \{ MD_s, C_i, D_i (MD_s, C_i)\} \tag{5}$$

Each node $j \in N_1(i)$ verifies the message MD_i by checking both digital signature D_i and D_s with the public key PUK_j and PUK_s stored in node j's neighbourhood certificate table. According to the principal of OPKM multi-hop certificate distribution, only the node $j \in N_1(i) \wedge j \notin N_1(s)$ will append its public key certificate C_j to the message MD_i and encrypts the whole contents with its private key PRK_j as its digital signature D_j (MD_i, C_j). Now the message is organised as a new message MD_j. Each node $j \in N_1(i) \wedge j \notin N_1(s)$ broadcasts the message MD_j to its 1-hop neighbours $N_1(j)$.

$$j \rightarrow N_1(j): \quad MD_j = \{ MD_i, C_j, D_j (MD_i, C_j) \} \quad (j \in N_1(i) \wedge j \notin N_1(s)) \tag{6}$$

Following node j's way, every other intermediate node continues the distribution process until the transferred public key certificate message reaches the destination node D.

During the process of the multi-hop certificate distribution, every participating intermediate node may collect other nodes' public key certificate information available in the exchanged certificate message to update its own network certificate table. The more certificate distribution processes a node joins, the more information the node may get about other nodes in the network. So OPKM multi-hop certificate distribution scheme encourage every node in the network to participate.

3) *Public Key Certificate Verification*

OPKM public key certificate verification is performed through both processes of neighbourhood certificate distribution and multi-hop certificate distribution.

In the process of ***neighbourhood certificate distribution***, the public key certificate verification is performed through neighbourhood monitoring. OPKM uses message broadcasting to distribute public key certificates among nodes in the network. Since a node can hear the message (and public key certificates, including its own) that its 1-hop neighbours re-broadcast, the node can verify that its public key certificate has been distributed correctly to both its 1-hop and 2-hop neighbours. If a node saw that its certificate was published incorrectly, then it will notify nodes in the network. For example, in Fig.1 the new node N has no direct contact with node F. The only way that node N can get to know node F's public key certificate is through node C. If node C wants to send a false public key certificate of node F in the message M_c to node N (e.g. in Step 2c of Fig. 1), such malicious behaviour can be easily detected by node F since both node F and node N receive the same message M_c at the same time.

If no neighbour node has any dispute about the public key certificate information broadcast by a node, the certificate information is deemed as having being distributed correctly. We believe that every node in the network wants its public key certificate to be distributed correctly via its 1-hop neighbours and it is within its own interests to detect any malicious modification of its certificate information. Such neighbourhood monitoring prevents a node from malicious modification through OPKM neighbourhood certificate distribution process.

In the process of ***multi-hop certificate distribution***, each intermediate node verifies that the received message has not been modified by the preceding two nodes. It can do so, since the preceding two nodes are its 1- and 2-hop neighbours, and therefore has their public key certificates. Once verified, the node signs the entire message and broadcasts it to the next nodes. If this check is applied by all nodes along the path, then any modifications will be identified.

For example, in Fig. 2 a public key certificate message is broadcast from source node *S* and transferred to node *D* via the intermediate nodes *B, C, N* and *O*.

The first three messages sent are illustrated in Fig. 3. Source node *S* sends the certificate message containing its own certificate, C_s, and the destination ID to node *B*. At point 1, node *B* verifies node *S*'s digital signature using the public key of node *S* obtaining from neighbourhood certificate table. Node *B* then broadcasts a new message containing C_s, C_b and (C_s, C_b) signed by node *B*. Upon receipt node *C* verifies node *B*'s digital signature and then node *S*'s digital signature. The verification of node *S*'s digital signature is performed to ensure node *B* has not modified the original message, i.e. C_s. This process of verifying the preceding two nodes digital signature continues, providing a chain of verification along the entire path.

4) Public Key Certificate Revocation

When a node believes that its public/private key pair is compromised or the validity period of its public key certificate is expired, the node will implement the public key certificate revocation process. For the case when node *i* wants to revoke its public key certificate, node *i* is required to broadcast a *PKC_RVK* revocation message MRK_i to all its neighbours $N_1(i)$. In message MRK_i, both node *i*'s old and new certificates (C_i and C_i') are included and signed with its new private key PRK_i'.

$$i \rightarrow N_1(i): \quad MRK_i = \{ PKC\_RVK, C_i, C_i', D_i(C_i, C_i') \}$$

Every node $j \in N_1(i)$ that receives message MRK_i will rebroadcast MRK_i to its one hop neighbours and update its network certificate repository and its neighbourhood certificate repository if node *i* is its neighbour within its two hops transmission.

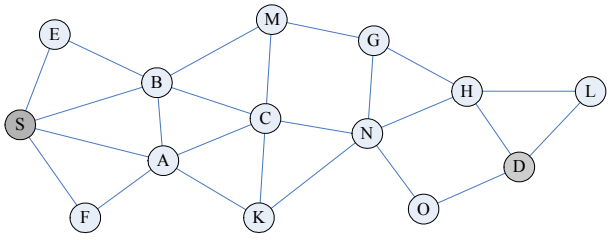

Fig. 2. An example ad hoc network

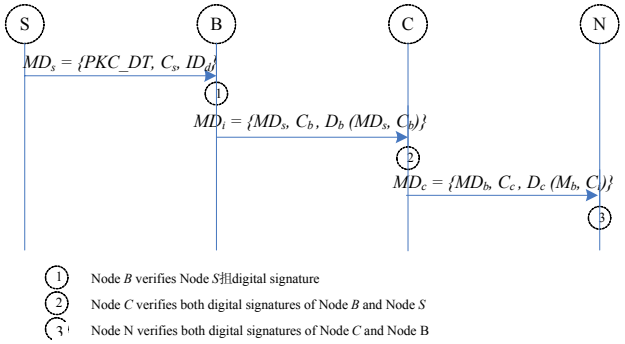

Fig. 3 Example verification process for multi-hop certificate distribution

IV. Security Discussion of Our Approach

Without relying on any central service and online trusted third party, the public key management for wireless ad hoc networks is vulnerable to man-in-the-middle attacks since multi-hop communications relies on intermediate nodes to relay the messages. To modify the message information, an intermediate node may implement man-in-the-middle attacks in three different ways: legitimate node attack, Impersonation attack and Sybil attack. Our proposed scheme of on-demand public key management for wireless ad hoc networks can defend these three kinds of man-in-the-middle attacks effectively in most cases.

A. Legitimate Node Attack

In this kind of man-in-the-middle attack, a malicious node joins the network as a legitimate node, but behaves maliciously intending to modify other's public key certificate information when it participates in the process of certificate distribution. Our approach can detect such malicious behaviour in both neighbourhood and multi-hop certificate distribution. During neighbourhood certificate distribution, any malicious modification on other neighbour's public key certificate will easily be detected through the neighbourhood monitoring as discussed in our certificate verification section. During the multi-hop certificate distribution, such malicious modification can be detected via a following intermediate node verifying digital signatures of its neighbours within two hops.

For example, in Fig.3 node *C* maliciously modifies the certificate information about node *S* in its message MD_s. Upon receiving the message MD_s, node *N* starts its verification as followings. First, node *N* verifies node *C*'s digital signature by decrypting it with node *C*'s public key PUK_c. Then node *N* verifies node *B*'s digital signature on the previous message MD_b with node *B*'s public key and applies the common hash function to the MD_s and C_b. If any information is modified inside the message, the value of the hash function cannot match. Consequently the malicious modification is detected.

B. Impersonation Attack

An Impersonation Attack involves a malicious node joining the network and masquerading as another legitimate node in the network. In this case, nodes wishing to exchange keys with node *X* may actually exchange keys with a malicious node *Y*, which is impersonating node *X*. Our scheme does not prevent such attacks outright, but can detect the attack in many cases. For example, as nodes store both neighbourhood and network certificates, including those collected during the multi-hop certificate distribution, nodes in the local neighbourhood of *Y* (impersonating node *X*) may already have the certificate of (the real) node *X*. Therefore, when node Y distributes the certificate pretending to be node *X*, 1- and 2-hop neighbours are highly likely to detect a problem (two different nodes saying they are *X*). The response to such an attack is area for future work.

C. Sybil Attack

A Sybil Attack [10] involves a malicious node using multiple fake identities to act as several nodes. By doing so, the malicious node may attempt to hide the modification of public key certificate information by the fact that the multiple fake nodes authenticated the information. Fig. 4(a) shows two honest nodes, A and C, and a malicious node B that is pretending to be three nodes, B_1, B_2 and B_3 (e.g. a laptop using multiple physical or virtual wireless interfaces). Both node *A* and *C* believe they each have three 1-hop neighbours (B_1, B_2 and B_3).

Fig. 4(b) shows an example of multi-hop certificate distribution from node *A* to node *C* using OPKM. The messages are passed via node *B*, which pretends to verify the certificate at (fake) nodes B_1, B_2 and B_3. B_1 can potentially modify the certificate, and since node *C* only verifies the previous 1- and 2- hops (i.e. B_3 and B_2) and B_1, B_2 and B_3 are in collusion, it cannot detect the modification. However, OPKM does not allow such a Sybil attack since, as outlined in Section III C, node *C* will disregard messages that arrive via two of its 1-hop neighbours (because of the potential of multiple paths a broadcast message will arrive from, a node only accepts those that arrive via a 2-hop neighbour and then a 1-hop neighbour). In the example, since node *C* believes B_1, B_2 and B_3 are three separate 1-hop neighbours, it drops the certificate distribution message, naturally expecting to receive the same message from node B_1, therefore avoiding the Sybil attack.

The approach to avoiding the Sybil attack given above can result in incorrect dropping of messages. For example, in Fig. 4(c) suppose instead we have three real (honest) nodes *B, D* and *E*. Node *C* has these three nodes listed as 1-hop neighbours. If however the links between nodes *B* and *C* and nodes *D* and *C* break (e.g. due to mobility), the message may pass as shown from *A-B-D-E-C*. In this case node *C* will incorrectly drop the message, since it has arrived from two of its 1-hop

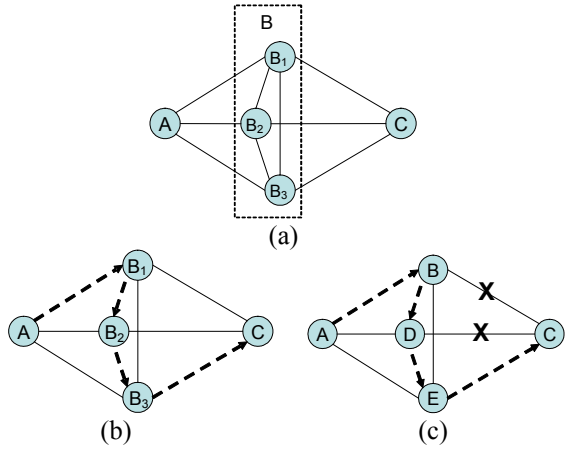

Fig. 4: Sybil attack examples

neighbours, nodes *D* and *E*. The likelihood of this occurring depends on the mobility patterns as well as the rate at which nodes maintain links with their neighbours (i.e. send out 'hello' messages). As a consequence, OPKM makes a trade-off to prevent the Sybil attack, at the expense of performance (i.e. dropped messages). Further investigation of this trade-off is planned for future work.

V. Discussion of Performance

Implementing security always comes at the expense of other features, in particular performance. In the previous sections we have proposed OPKM and informally discussed how it can prevent certain security attacks. Below we discuss the implications of OPKM on network performance in three key areas. Detailed performance analysis is intended for future work.

A. Overhead of Asymmetric Key Encryption

The use of asymmetric (public) key encryption introduces computational overhead at each mobile node. Although this may impact resources at the node (e.g. CPU, battery), as well as delays in processing, it is reasonable to assume users requiring a secure network are willing to make this performance trade-off. If necessary, enhancements such as shortening encryption key lengths may be made without having adverse affect on the operation of OPKM.

B. Scalability of Certificate Tables

In OPKM each node maintains a neighbourhood certificate and network certificate table. Given certificates are of the order of 100's of bytes [11], the size of these tables should pose little or no constraints on memory limited devices (e.g. when the network contains 1000 nodes). To cope with the dynamic nature of the ad hoc network, a soft state approach to storing the tables should be used. The keys of nodes should become stale after timeouts and/or lack of observed traffic to that node. Optimising these parameters is a matter for future work.

C. Reliance on Broadcast

Currently we have assumed use of a blind broadcast in the neighbourhood and network certificate distribution processes. However, in CSMA/CA networks such broadcasting is particularly expensive in terms of redundant packets, probability of collision and wireless medium congestion, especially when the density of neighbourhood increases [12]. Therefore, for OPKM to be scalable, optimised broadcasting schemes may be considered (e.g.[13] [14]). One possible approach is for nodes to use the knowledge of their 1- and 2-hop neighbours, i.e. a node re-broadcasts only when it's 1-hop neighbours differ from the previous node. It should be noted that to a large extent OPKM is independent of the specific optimised broadcasting scheme used. As future work we will analyse the performance and scalability of OPKM for different broadcasting schemes.

VI. Conclusion and Future Work

Securing wireless ad hoc networks is a necessary step in deploying future ubiquitous mobile services. Public key cryptography is a natural choice for establishing trust in such dynamic and open networks. In this paper we have presented the design of a novel public key management scheme for wireless ad hoc networks. By utilising the simple fact that broadcasting public key certificates to neighbours, nodes can efficiently gather certificates of their 1- and 2-hop neighbours, and that a chain of trust can be established across the entire network, OPKM is able to provide key management on demand. OPKM can prevent and/or detect various man-in-the-middle attacks. To our best knowledge, OPKM is the first approach to implement public key management on demand in self-organising wireless ad hoc networks. The main contributions of OPKM are:

- Able to provide public key management service on demand for exchanging the certificate information through multi-hop communication;
- Can be operated in a fully self-organised manner by nodes themselves without relying on any online trust third party;
- Flexible to adapt to dynamic changes of neighbour relationship and network membership caused by node movement;
- No threshold limitation as in the distributed approaches with threshold cryptography.

OPKM is a new public key management scheme for wireless ad hoc networks. This paper proposes the basic idea for OPKM. As the next steps for developing OPKM we plan to:

- Formalise the protocol and verify its correctness under specific types of attacks.
- Analyse the performance of OPKM in terms of overhead introduced into the network. This includes analysing the scalability of OPKM and optimising performance using different broadcast mechanisms.
- Enhance OPKM with more efficient certificate conflict resolution mechanisms and possible integration with routing protocols (e.g. AODV).
- Analyse the effectiveness of OPKM in terms of preventing certain attacks. For example, successful prevention of some attacks depends on nodes having learnt sufficient knowledge/trust about other nodes – we intend to determine under what circumstances does this hold.
- Given the above (i.e. that using OPKM there may be cases when attacks cannot be prevented), investigate compatible schemes for intrusion detection and response.

Reference

[1] L. Buttyan and J.-P. Hubaux, "Report on a working session on security in wireless ad hoc networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, pp. 74-94, 2003.

[2] I. Chlamtac, M. Conti, and J. J.-N. Liu, "Mobile Ad hoc Networking: Imperatives and Challenges," *Ad Hoc Network Journal*, vol. 1, pp. 13-64, 2003.

[3] W. Diffie and M. E. Hellman, "New Directions in Cryptography," in *IEEE Transactions on Information Theory*, vol. 22, 1976, pp. 644-654.

[4] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network*, pp. 24-30, 1999.

[5] A. Shamir, "How to share a secret," in *Communications of the ACM*, 1979, pp. 612-613.

[6] S. Yi and R. Kravets, "MOCA: Mobile Certificate Authority for Wireless Ad Hoc Networks," Proceedings of 2nd Annual PKI Research Workshop Program (PKI 03), Gaithersburg, Maryland, USA, 2003.

[7] A. Khalili, J. Katz, and W. A. Arbaugh, "Toward Secure Key Distribution in Truly Ad-hoc Networks," Proceedings of 2003 Symposium on Applications and the Internet Workshops (SAINT 2003 Workshops), Orlando, FL, USA, 2003.

[8] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks," Proceedings of 2001 International Conference on Network Protocols ICNP, Riverside, CA, 2001.

[9] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," *IEEE Transactions On Mobile Computing*, vol. 2, pp. 52-64, 2003.

[10] J. R. Douceur, "The Sybil Attack," Proceedings of 1st International Workshop on Peer-to-Peer Systems (IPTPS '02), Cambridge, MA, USA, 2002.

[11] A. Young and M. Yung, "Hash to the Rescue: Space Minimization for PKI Directories," Proceedings of Information Security and Cryptology - ICISC 2000, (Lecture Notes in Computer Science Vol.2015), Seoul, South Korea, 2000.

[12] J. Lipman, P. Boustead, J. Chichara, and J. Judge, "Optimised Flooding Algorithms for Ad hoc Networks," Proceedings of The 2nd Workshop on the Internet, Telecommunications and Signal Processing (WITSP'03), Coolangatta, Gold Coast, Australia, 2003.

[13] W. Peng and X.-C. Lu, "On the reduction of broadcast redundancy in mobile ad hoc networks," Proceedings of First Annual Workshop on Mobile and Ad Hoc Networking and Computing 2000, Boston, MA, USA, 2000.

[14] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint relaying for flooding broadcast messages in mobile wireless networks," Proceedings of the 35th Annual Hawaii International Conference on System Sciences, Los Alamitos, CA, USA, 2002.