Mobility Helps Security in Ad Hoc Networks*

Srdjan Čapkun, Jean-Pierre Hubaux, and Levente Buttyán†
Laboratory for Computer Communications and Applications (LCA)
School of Information and Communication Sciences (I&C)
Swiss Federal Institute of Technology Lausanne (EPFL)
CH-1015 Lausanne, Switzerland
srdan.capkun@epfl.ch, jean-pierre.hubaux@epfl.ch, buttyan@hit.bme.hu

ABSTRACT

Contrary to the common belief that mobility makes security more difficult to achieve, we show that node mobility can, in fact, be useful to provide security in ad hoc networks. We propose a technique in which security associations between nodes are established, when they are in the vicinity of each other, by exchanging appropriate cryptographic material. We show that this technique is generic, by explaining its application to fully self-organized ad hoc networks and to ad hoc networks placed under an (off-line) authority. We also propose an extension of this basic mechanism, in which a security association can be established with the help of a "friend". We show that our mechanism can work in any network configuration and that the time necessary to set up the security associations is strongly influenced by several factors, including the size of the deployment area, the mobility patterns, and the number of friends; we provide a detailed investigation of this influence.

Categories and Subject Descriptors

C.2.0 [**Computer Systems Organization**]: Computer - Communication Networks—*General*

General Terms

Security

*The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322 (`http://www.terminodes.org`).

†Now with Budapest University of Technology and Economics Department of Telecommunications Magyar tudosok krt. 2, H-1117 Budapest, Hungary.

MobiHoc'03, June 1–3, 2003, Annapolis, Maryland, USA.

Keywords

Ad Hoc Networks, Mobility, Security associations

1. INTRODUCTION

Security and mobility seem to be at odds with each other. Security is usually enforced by a static, central authority that is generally in charge of securing the system under consideration, be it a communication network, an operating system, or the access system to the vault of a bank. In this case, because users are static as well, their locations are predictable, they are more likely to be available, and the system can more easily perform appropriate controls.

In this paper, we will show that this intuition can be misleading: mobility, far from being a hurdle, can be useful to establish security associations[1] between any two mobile nodes of a given network.

The idea underpinning the solution that we propose is extremely straightforward, as it simply mimics human behavior: if people want to communicate securely, they just get close to each other in order to exchange information and to establish (or reinforce) mutual credentials.

In spite of its simplicity, this idea is very powerful, as it can be applied to virtually *any* mobile ad hoc network at *any* layer (from the MAC up to the application layer). It makes it possible to provide security either without any kind of central authority, or with an authority the role of which is limited to the initial delivery of certificates.

In the first case, we will consider *fully self-organized* security: in such a setting, there is no central authority whatsoever, and the establishment (and releasing) of security associations is purely based on mutual agreement between users. In this case, we will assume the presence of a secure side channel (e.g., physical contact between nodes or infrared communication) which can be established at will by two users close to each other. We will show that our solution is extremely intuitive for the users, as it faithfully matches human relationships and encounters.

The second case corresponds to situations where an authority provides the authorization to each mobile node to join the network, but it does so only at the initialization of each node (the authority is considered to be off-line); moreover, we allow the nodes to join the network at different times (in general, the authority does not even know how many nodes are eventually going to be present in the

[1]The precise definition of security association will be provided in Section 3, as it depends on the considered scenario.

network). An important usage of this approach is to secure routing. Usually, the security of routing in mobile ad hoc networks requires a rather heavy initialization phase, which can even include the pre-loading in nodes of secret keys shared with other nodes [16, 15, 23]. This is well suited for "closed" networks, but not for the case we consider here; in particular, it would prohibit the entry of new nodes once the network is already in operation.

As we will see, the proposed system is "democratic", in the sense that each node is assigned exactly the same role, in contrast to some other proposals [27]. Moreover, our solution avoids the weakness of having a single point of failure (except the off-line certification authority, in the case in which there is one).

To the best of our knowledge, the only research published so far on this topic [2] presents the idea that mobility can be used to set up security associations. Here, we significantly develop this concept by quantifying the benefits of mobility and by introducing a new mechanism ("friends") that supports the building of security associations, even between nodes that do not meet physically.

The work presented in this paper is a part of the Terminodes Project [14].

The organization of the paper is the following. In Section 2, we survey the related work. In Section 3, we provide the model of our system and describe how security associations are created based on encounters. In Section 4, we study the pace at which the security associations are created. Finally, we conclude the paper in Section 5.

2. STATE OF THE ART

Several solutions have already been proposed to set up security associations between nodes in an ad hoc network.

In [27], Zhou and Haas propose a distributed public-key management service for ad hoc networks. The service, as a whole, has a public/private key pair K/k, which is used to verify/sign public-key certificates of the network nodes. It is assumed that all nodes in the system know the public key K and trust any certificate signed with the private key k. The private key k is not known to any of the nodes. Instead, it is divided into n shares using a $(n, t+1)$ *threshold cryptography* scheme, and the shares are assigned to n arbitrarily chosen nodes, called servers. For the service to sign a certificate, each server generates a partial signature for the certificate using its private key share and submits the partial signature to a combiner that computes the signature from the partial signatures. The application of threshold cryptography ensures that the system can tolerate a certain number $t < n$ of compromised servers in the sense that at least $t + 1$ partial signatures are needed to compute a correct signature. Besides threshold signatures, the proposed key management service also employs *proactive share refreshing* in order to tolerate *mobile adversaries* and to adapt its configuration to changes in the network. Unfortunately, the proposal has two major drawbacks: First, it requires an authority to empower the servers. Second, it assumes that some of the nodes must behave as servers, which does not seem to be realistic, at least in civilian applications.

A more recent proposal by Kong et al. [17] describes a similar approach, but it provides a more fair distribution of the burden by allowing *any* node to carry a share of the private key of the service. The advantage is increased availability, because now *any* $t + 1$ node in the local neighborhood of the requesting node can issue or renew a certificate. Another interesting novelty is that any node not yet possessing a share can obtain a share from any group of at least $t + 1$ nodes that already possess a share. It is therefore sufficient to initialize only $t + 1$ nodes with appropriate shares when the system is initialized. But, as in [27], the first $t+1$ nodes must be initialized by a trusted authority. In addition to this drawback, there are two problems with this proposal: First, the number t must be a trade-off between availability and robustness; but it is unclear how the value of t can be changed in case the overall number of nodes significantly increases (or decreases). Second, the system seems to be vulnerable to the Sybil attack [9]: an attacker can take as many identities as necessary to collect enough shares and reconstruct the system's private key.

A different approach by Asokan and Ginzboorg in [1] is based on a shared password. As mentioned by the authors, nodes willing to establish a secure session must share a *prior context*. In the considered scenario, a small group of people get together in a room for an ad hoc meeting and want to set up a wireless network session among their laptop computers for the duration of the meeting. It is assumed that they do not have access to public-key infrastructure or third-party key management services. The proposed solution is the following: A fresh password is chosen and shared among those present in the room (e.g., by writing it on a blackboard). However, it would be a mistake to use this password directly as the shared key, as the protocol would then be vulnerable to *dictionary attacks* [20]. Therefore, the authors propose to make use of *password-authenticated key exchange* by which the parties derive a strong shared key starting from only a weak secret (i.e., the password). This proposal only works if the parties can share a password by being physically present in the same room. Moreover, it has the drawback of being somewhat cumbersome, as it requires the users to type the password in their personal device. Finally, the system can easily be penetrated by an adversary who could obtain access to the password by means of a microphone or a camera.

Another approach, originally designed for the address ownership problem in Mobile IPv6, is described by Montenegro and Castelluccia in [21] and by O'Shea and Roe in [22]. Their idea is to derive the IP address of the node from its public key: first, the public key is hashed with a cryptographic hash function, and then, (part of) the hash value is used as part of the IP address of the node. The advantage is that there is no longer need for certificates that bind the node's address to its public key, since one is derived from the other in a cryptographically verifiable way. In our proposal, we adopted this approach to bind node addresses to public keys, and we also considered the problem of binding user identifiers to public keys.

In [8], we propose a self-organized public-key management system for ad hoc networks, which is similar to PGP in the sense that users issue certificates for each other based on their personal acquaintances. In that system, each user maintains a *local certificate repository*. When two users want to verify the public keys of each other, they merge their local certificate repositories and try to find (within the merged repository) appropriate certificate chains that make the verification possible. We propose several construction algorithms and show that even simple algorithms can achieve high performances in the sense that any user can find at least one certificate chain to any other user in their merged

repository with a high probability, even if the size of their local repositories is kept small. There are, however, two disadvantages: First, each user is required to build her local certificate repository before she can use the system; this leads to some overhead, both in terms of time and bandwidth. Second, as any approach that uses certificate chains, this approach also assumes that trust is transitive. In order to alleviate the latter problem, we rely on local detection of inconsistent certificates and the use of authentication metrics.

As we mentioned earlier, an approach conceptually very close to our proposal is described in [2]. In that work, Balfanz et al. assume (as we do in this paper) that the users can make use of privileged side channels, they call *location-limited channels*, for the establishment of security associations when they are in the vicinity of each other. The location limited channels are assumed to be secure against active attacks. However, the authors do not explain in the paper how the mobility of the nodes can be used to progressively reinforce the security of the nodes. Furthermore, it does not contain a mechanism based on a common trusted friend.

Finally, we must mention the work of Grossglauser and Tse in [11], and Grossglauser and Vetterli in [12], the first papers, to our knowledge, claiming that mobility can be useful in solving certain problems of ad hoc networks. In [11], the authors study a model of an ad hoc network where nodes are assumed to be mobile and communicate in random source-destination pairs. They examine the per-session throughput for applications with loose delay constraints, such that the topology changes over the time-scale of packet delivery. They show that under this assumption, the per-user throughput can increase dramatically when nodes are mobile rather than static. In [12], the authors show that node mobility can be used to disseminate destination location information in ad hoc networks without incurring any communication overhead. They achieve this by letting each node maintain a local database of the time and location of its last encounter with other nodes in the network. This database is consulted by packets to obtain estimates of their destination's current location. As a packet travels towards its destination, it is able to successively refine an estimate of the destination's precise location, because node mobility has "diffused" estimates of that location. A more recent work by Dubois-Ferriere, Grossglauser and Vetterli [10] shows that if nodes keep track of their encounters, route discovery can be performed at a much lower cost than with traditional broadcast search methods.

A more detailed overview of the security issues in wireless ad hoc networks can be found in [6].

More work in the area of ad hoc network security has been reported in [26, 25, 18, 4, 24, 13, 3, 5]; these papers are loosely related to the security problems observed in this paper.

3. SYSTEM MODEL

3.1 Assumptions and overview

We consider and discuss two models: the first is fully self-organized, and the second assumes the presence of a trusted authority.

3.1.1 Fully self-organized ad hoc networks

We consider an ad hoc network of mobile *nodes*, where each node represents a *user* equipped with a personal mobile *device*. We assume that each legitimate user has a single device.

We consider the network to be *fully* self-organized, meaning that there is no infrastructure (hence no PKI), no central authority, no centralized trusted third party, no central server, and no secret share dealer *even in the initialization phase*. As already mentioned, a fundamental assumption is that each node is its own authority domain. To make the problem tractable, we therefore assume that each node is able to generate cryptographic keys, to check signatures and, more generally, to accomplish any task required to secure its communications (including to agree on cryptographic protocols with other nodes).

If a user i can relate the name (or the face) of another user j to his (j's) public key, we will say that there is a *one-way security association* from i to j. Two one-way security associations between i and j (one in each direction) constitute a two-way security association between i and j; in the rest of the paper, security associations will be considered to be two-way if not mentioned otherwise.

When they meet, users are naturally given the possibility to visually identify each other. The decision to set up a security association between two nodes is based on this physical encounter. To support this mechanism, we assume that each device is equipped with a short range connectivity system (e.g., infrared or wire). We call a channel established by this mechanism a *secure side channel*. A secure side channel can only be point to point and works only when the nodes are within a "secure range" of each other. We consider this assumption to be realistic, as virtually all personal mobile devices are equipped with infrared interfaces.

The secure side channel is used to set up security associations between nodes by exchanging cryptographic material. We assume that the activation of the side channel is made by both users consciously and simultaneously. The users are given the opportunity to associate a "human face" to the established security association. This operation is very similar to the exchange of business cards; in fact, it can even be transparently combined with the exchange of *electronic* business cards (e.g., exchange of vCards[2] between PDAs). If a user wants to establish a security association with a user-independent device (e.g., a printer), she will identify the device visually and bind its identity to the context in which the device operates. In this paper, however, we focus on the establishment of security associations between users' personal communication devices.

We assume that an adversary can eavesdrop on all radio links and can manipulate messages in all kinds of ways. However, the adversary cannot modify messages transmitted over the secure side channel. Note that we do not require the secure side channel to protect the confidentiality of exchanged information. Finally, we consider that an adversary can have at its disposal as many fake devices as it wants.

Our proposal is based on public-key cryptography for two main reasons. First, the distribution of public keys does not require confidential channels. In our case, this means that the secure side channel does not need to provide protection against eavesdropping, and therefore its implementation can

[2]http://www.imc.org/pdi/

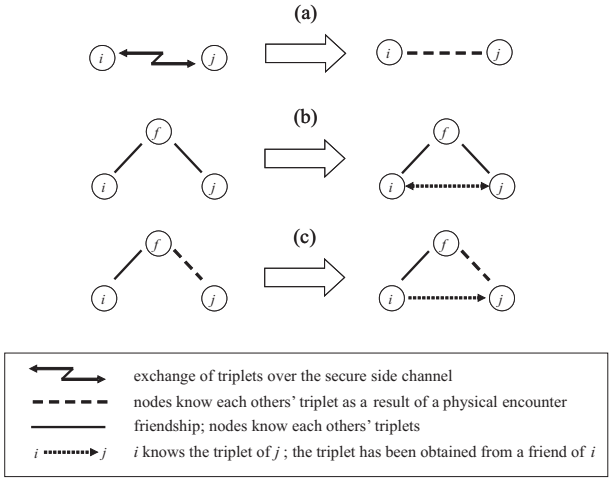

Figure 1: Three mechanisms to create new security associations using (a) the secure side channel, (b) a common friend, and (c) the combination of the first two approaches.

be based on various technologies that would not be suitable otherwise for the exchange of symmetric keys. Second, as we will see, we allow the nodes to establish security associations with the help of *friends*. Friends can distribute public-keys (e.g., in form of certificates) without being able to decrypt messages that are encrypted with those public keys and to forge signatures that can successfully be verified with those public keys. On the other hand, if two nodes exchanged a symmetric key via a common friend, then the friend would be able to use that symmetric key for encrypting and decrypting subsequent messages of the two nodes; we consider this to be undesirable; our system does not allow it.

In order to support different applications, we allow each node to have several public key / private key pairs, that are all generated by the node itself. However, in order to make the presentation easier in the rest of the paper, we will assume that each node has a single public key / private key pair. Our proposal works without this assumption too, and we will briefly point out at the appropriate places how it can incorporate multiple key pairs per node.

Each node must have a node address that is used by the routing protocol. We assume that the node address is generated from the public key of the node[3] by making use of a technique similar to CAM [22] or SUCV [21]. In this way, node addresses become verifiable. Note that a malicious node may generate several node addresses for itself and freely distribute them to other nodes. Whether this is a problem very much depends on how the routing protocol is secured. A thorough study of this issue is left for future work.

We assume that the user identification is human-friendly; typically, a user is identified by his or her (first name, last name) pair. This is often considered to be insufficient, because of homonyms. However, as we will see later, the identity of a given user is used exclusively via a common friend; therefore, the context in which the operation takes place will naturally remove any potential ambiguities (for example, when asked by someone to provide the address or phone number of a friend, we very rarely face the problem of homonyms, and we easily cope with it). Peoples' names change only exceptionally, meaning that users can rely on them in the long run.

The basic mechanism for setting up security associations relies on the physical encounters of users and the activation of the secure side channel. However, in order to expedite the process, we assume that nodes can also rely on *friends*. Two nodes i and j are said to be friends if (i) they trust each other to always provide correct information about themselves and about other nodes they have previously encountered, and (ii) they have already established a security association between each other (typically, they know each others' public keys). The security association between friends is assumed to be established (or at least checked) over an out-of-band channel. We assume the friend relationship to be non-transitive.

Even if in our approach the nodes establish security associations through encounters, we do allow users to establish security association by other means (e.g., through certificates issued by a trusted authority or through off-line channels). Thus, our scheme can support and enhance existing security solutions for ad hoc networks (e.g., the creation of a certificate graph [8]).

3.1.2 Ad hoc networks with a central authority

Here, we consider an ad hoc network of mobile *nodes*, controlled by an (off-line) central authority. This means that the authority controls network membership and decides which nodes can join the network and how.

We assume that each node has a unique identity (e.g., assigned to it by the authority). Furthermore, each node holds a certificate signed to it by the authority that binds the node's identity and its public-key. We further assume that each node holds a correct public key of the authority, so that it can verify the correctness of the certificates that other nodes hold. Here, like in the self-organized approach, each node is able to generate cryptographic keys, to check signatures, and more generally to accomplish any task required to secure its communications (including to agree on cryptographic protocols with other nodes).

If a node i possesses a certificate signed by the central authority that binds node j with its (j's) public key, then we say that there exists a one-way security association from i to j. Here again, two one-way security associations between nodes i and j (one in each direction) constitute a two-way security association between the nodes.

When two nodes move into power range of each other, they will exchange certificates that contain their public keys and establish a security association. This direct establishment of security associations breaks the routing-security interdependence cycle [16]: Security associations cannot be established over multiple hops as the routing protocol does not operate securely (because security associations are not established yet). This means that if two users want to establish security associations, their packets could be sent through false routes, or simply dropped. Some solutions to the routing-security interdependence problem were proposed by Hu, Perrig and Johnson [16]. Their first proposed solution consists in pre-loading pairwise keys in all nodes to create all the security associations at the initialization. However, this approach prevents the insertion of new nodes in

[3]If the node has several public keys, then the node address is generated from a designated one.

the network. The second approach makes use of an on-line key distribution center. Although effective, this approach requires a costly initialization phase and the use of complex security protocols. Our mobility-based approach is different in the sense that it enables a flexible setup of security associations, simplifies the introduction of new nodes in the network, and requires only an off-line authority; the drawback is that the establishment of the security associations requires some time, as detailed in Section 4.

Here, again, like in the self-organized approach, we allow the use of the friends mechanism to facilitate the establishment of the security associations. We note, however, that in some lower layer applications (e.g., routing), the friends mechanism will not have practical value.

The major difference between the fully self-organized and the authority-based approach stands in user involvement: In a fully self-organized approach, users need to establish security associations consciously, while in the authority-based approach, users do not need to be aware of the establishment of the security associations, as it is done automatically. The use of either of these approaches will strongly depend on the purpose of the network. Typically, the self-organized approach will be useful in securing personal communications on the application level, whereas the authority-based approach will be used to secure networking mechanisms such as routing.

For simplicity, in the rest of the paper, we describe the operation of our scheme with the fully self-organized approach; we note, however, that the presented analysis equally applies to the authority-based approach.

3.2 Mechanisms to establish security associations

A (two-way) security association between two nodes i and j is represented by the triplet $(u_i,\ k_i,\ a_i)$ at the side of j and the triplet $(u_j,\ k_j,\ a_j)$ at the side of i, where u_i and u_j are the names of the users that are associated with nodes i and j; k_i and k_j are the public keys of nodes i and j; and a_i and a_j are the node addresses of i and j, respectively[4]. Given a security association between nodes i and j, they can verify that the node addresses match the public keys, and they can set up a secure communication channel between themselves, which protects the integrity and confidentiality of the exchanged messages. In fact, for efficiency reasons, i and j may want to use symmetric key cryptography for the protection of their messages; in this case, the symmetric keys are established using the public keys in the security association.

If only one of the nodes, say i, has the triplet $(u_j,\ k_j,\ a_j)$, then there is a one-way security association between nodes i and j. In this case, i can check the node address of j, verify messages signed by j, and send encrypted messages to j that j can decrypt, but j cannot check the node address of i, verify messages signed by i, nor send encrypted messages to i that i can decrypt. Even so, there are applications in which one-way security associations may be used. Note that two one-way security associations between two nodes (one association in each direction) constitute a (two-way) security association; therefore, security associations can be established in two steps by setting up a one-way security association in each step.

[4]If the nodes have several public keys, then instead of a single public key, the triplets contain a set of public keys.

There are three mechanisms that support the establishment of new security associations; they are illustrated in Figure 1. The first is when two nodes i and j are in the vicinity of each other: they can exchange their triplets using the secure side channel. Since the secure side channel ensures the integrity of the exchanged messages, it precludes the possibility of a man-in-the-middle attack, and therefore, it ensures the secure binding of the received name, public key, and address. In addition, the users can easily verify the validity of the received names because the names should correspond to the people present at the encounters. Each node can also verify that the other node indeed possesses the private key that belongs to the received public key by executing a simple challenge-response protocol. Finally, the node addresses can be verified with the public keys.

With the second mechanism, two nodes i and j can exchange their triplets if they have a common friend f. A simple solution is the following: Since f knows the triplets of both i and j, it can issue (on request from i and/or j) fresh certificates for both triplets and send them to j and i, respectively, via the network. Both i and j know the public key of f and they also trust f, therefore they can both verify the received certificates and will accept the information therein if the verification is successful.

The third mechanism establishes only a one-way security association between two users i and j, and is based on a combination of the friendship relationships and the encounters. If nodes i and f are friends and f has obtained the triplet of j in an encounter with j, then f can issue (on request from i) a fresh certificate for the triplet of j, and send this certificate to i via the network. Since i knows the public key of f and also trusts f, she can verify the received certificate and accept the received triplet if the verification is successful.

The second and the third mechanisms show the usefulness of the friends mechanism for the establishment of security associations. Note that due to the friends mechanism, a new node that joins the network will be able to communicate almost instantaneously with all the nodes with which its friends already established security associations.

The proposed mechanisms avoid trust transitivity (it is allowed only in the "first hop" through a direct friend). As shown on Figure 1, friend assisted establishment can be performed only if a friend has a directly established security associations with a target node. This restriction prevents nodes from establishing security associations through a friend's friend, or through a friend of a friend of a friend, etc.

There are many ways to implement these mechanisms; here we propose two protocols as examples for possible implementations. The first mechanism described above can be built on Protocol 1; Protocol 2 can be used to implement the second and third mechanism.

Protocol 1: Direct Establishment of a Security Association

msg1	$i \rightarrow j:$	$r_i \mid u_i \mid k_i \mid a_i$
msg2	$j \rightarrow i:$	$r_j \mid u_j \mid k_j \mid a_j$
	$i:$	$u_j?;\ match(k_j, a_j)?$
	$j:$	$u_i?;\ match(k_i, a_i)?$
msg3	$i \rightarrow j:$	$\sigma_i(r_j \mid u_i \mid u_j)$
msg4	$j \rightarrow i:$	$\sigma_j(r_i \mid u_j \mid u_i)$

Protocol 1 is executed entirely through the secure side

channel. First, i and j exchange their triplets along with two freshly generated random numbers r_i and r_j. Then, i and j verify locally that the received names u_j and u_i, respectively, correspond to the person they are facing, and that the received addresses a_j and a_i match the received public keys k_j and k_i, respectively. If the verifications are successful, then i and j exchange their signatures $\sigma_i(r_j \mid u_i \mid u_j)$ and $\sigma_j(r_i \mid u_j \mid u_i)$ on the received random number and the received names. If i and j can successfully verify the signatures with the received public keys, then both can be sure that the other node knows the corresponding private key.

Depending on the implementation of the secure side channel, it may be impractical to execute the entire protocol through the secure side channel. For example, if the users use an infrared link as the side channel, it might be problematic for the users to point at each others' devices during all time necessary for protocol execution. Therefore, we propose a variant of Protocol 1 where only two short messages are exchanged through the secure side channel, and the rest of the messages are sent using the radio interface of the devices:

Protocol 1': Direct Establishment of a Security Association (variant)		
msg1 (ssch.)	$i \rightarrow j:$	$a_i \mid \xi_i = h(r_i \mid u_i \mid k_i \mid a_i)$
msg2 (ssch.)	$j \rightarrow i:$	$a_j \mid \xi_j = h(r_j \mid u_j \mid k_j \mid a_j)$
msg3 (radio ch.)	$i \rightarrow j:$	$r_i \mid u_i \mid k_i \mid a_i$
msg4 (radio ch.)	$j \rightarrow i:$	$r_j \mid u_j \mid k_j \mid a_j$
$i:$	$h(r_j \mid u_j \mid k_j \mid a_j) = \xi_j?;\ u_j?;\ match(k_j, a_j)?$	
$j:$	$h(r_i \mid u_i \mid k_i \mid a_i) = \xi_i?;\ u_i?;\ match(k_i, a_i)?$	
msg5 (radio ch.)	$i \rightarrow j:$	$\sigma_i(r_j \mid u_i \mid u_j)$
msg6 (radio ch.)	$j \rightarrow i:$	$\sigma_j(r_i \mid u_j \mid u_i)$

In Protocol 1', i and j first exchange, through the secure side channel (ssch.), their addresses a_i and a_j and the cryptographic hash values $\xi_i = h(r_i \mid u_i \mid k_i \mid a_i)$ and $\xi_j = h(r_j \mid u_j \mid k_j \mid a_j)$ of their random numbers and triplets. After this initial exchange, i and j can send messages to each other through the radio interface since they have exchanged their addresses. The remaining messages are the same as the messages of Protocol 1. However, after the exchange of message 3 and 4, i and j also verify if the hash value of the received random number and triplet is equal to the received hash values ξ_j and ξ_i, respectively. If so, then they can be sure that they have received the random number and the triplet from the party with which they exchanged the first messages through the secure side channel.

Protocol 2: Friend-Assisted Establishment of a Security Association		
msg1	$i \rightarrow f:$	$req : u_j \mid r_i$
msg2	$f \rightarrow i:$	$u_j \mid k_j \mid a_j \mid \sigma_f(r_i \mid u_j \mid k_j \mid a_j)$

In Protocol 2, i requests the triplet of j from a friend f by sending the name u_j and a freshly generated random number r_i to f. User f responds with a message that contains the requested triplet and a signature $\sigma_f(r_i \mid u_j \mid k_j \mid a_j)$ on the received random number and the triplet itself. If the signature verifies correctly, then i accepts the triplet. Using Protocol 2 to establish a one-way security association between i and j (third mechanism described above) is straightforward. To set up a (two-way) security association, Protocol 2 should be executed by i and f and by j and f too.

4. PERFORMANCE EVALUATION

In this section, we provide an estimate of the pace at which security associations are created. We assume that initially each node establishes security associations only with its friends; we further assume that each node has the same number of friends.

4.1 Terminology

Let the population of nodes be represented by the set N, with cardinality $|N| = n$. We will designate by matrix F the friend relationships between nodes at the beginning of the operation of the network; the matrix P designates the desired status of security associations; and the matrix $E(t)$ designates the status of security associations at time t. All three matrices are of size $n \times n$. For the sake of generality, we assume F to be non-symmetric. Note that none of the matrices is stored as such; In reality, each node contains a row of each of them.

More specifically, matrix $F = [f_{ij}]$ is defined as follows:

$$f_{ij} = \begin{cases} 1 & \text{if } i \text{ trusts } j \text{ (i.e., } j \text{ is a friend of } i) \\ 0 & \text{otherwise} \end{cases}$$

Matrix $P = [p_{ij}]$ captures the fact that in the general case, a given user is interested in communicating with only a subset of the other users:

$$p_{ij} = \begin{cases} 1 & \text{if } i \text{ wants to know the public key} \\ & \text{and node address of node } j \\ 0 & \text{otherwise} \end{cases}$$

Finally, $E(t) = [e_{ij}(t)]$ represents the evolution of the security associations over time:

$$e_{ij}(t) = \begin{cases} 1 & \text{if, at time } t, i \text{ knows the public key} \\ & \text{and node address of node } j \\ 0 & \text{otherwise} \end{cases}$$

At the beginning, we will have $E(t_0) = F$. Over time, the number of "1"s in $E(t)$ will increase, until they represent a "superset" of the "1"s contained in P, at which stage we will consider that the system has achieved the required level of security. In other words, if we represent this time by t_ω, we will have that $p_{ij} = 1$ implies $e_{ij}(t_\omega) = 1$ for all $1 \leq i, j \leq |N|$.

Regardless of the kind of the cryptosystem used, we should note that it is possible to use friendships even before the nodes start moving: if $f_{ik} = 1$ and $f_{kj} = 1$, then $e_{ij}(t_0)$ and $e_{ji}(t_0)$ can be set to 1.

The speed of the creation of security associations depends on the likelihood that the nodes will be in the vicinity of each other. For this reason, the time needed for the nodes to establish security associations of interest is strongly related to the kind of mobility patterns that the nodes follow and the number of the security associations they want to establish. Therefore, we study the evolution of the matrix $E(t)$ under two different mobility models: random walk and (restricted) random waypoint.

In our analysis, we will observe two values: the convergence $r(t)$, which represents the fraction of the required security associations established until time t, and the convergence time t_ω, which is the time needed to establish all the

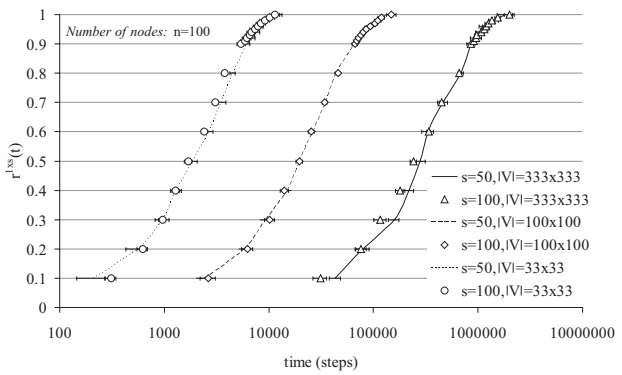

Figure 2: Convergence $r^{1\times s}(t)$ with the random walk model for various sizes (number of vertices $|V|$) of the rectangle grid and with different numbers of desired security associations s.

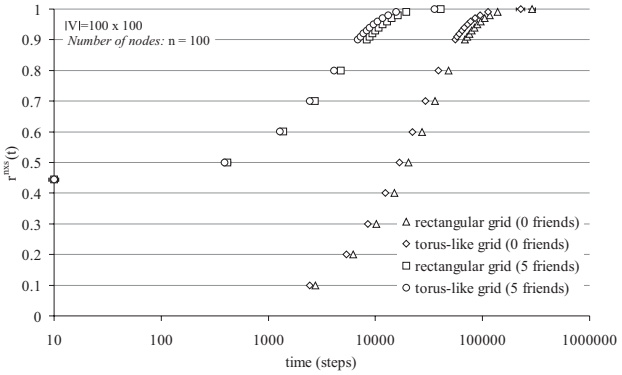

Figure 3: Convergence $r^{n\times s}(t)$ with the random walk model on 100×100 size rectangular and torus-like grids for $n = 100$ and $s = 99$, with and without the friend mechanism.

desired security associations. $r(t)$ is thus computed as

$$r(t) = \frac{\sum_{i,j}^{n} e_{ij}(t) \cdot p_{ij}}{\sum_{i,j}^{n} p_{ij}}$$

and the convergence time t_ω is the earliest time at which $r(t_\omega) = 1$.

We consider two scenarios: (i) a single node wants to establish s security associations and (ii) all n nodes want to establish security associations with s nodes each. We denote the convergence and the convergence time in the first case by $r^{1\times s}(t)$ and $t_\omega^{1\times s}$, and in the second case by $r^{n\times s}(t)$ and $t_\omega^{n\times s}$, respectively.

4.2 Random walk mobility

Although in reality, node positions are continuous processes in continuous time, here we use a discrete approximation and we assume two types of topologies: a rectangular grid and a torus-like grid. A rectangular grid of size m^2 is a two-dimensional square grid of $m \times m$ vertices with boundaries. A torus-like grid of size m^2 is a two-dimensional square grid of $m \times m$ vertices with continuous boundary conditions, meaning that its boundary vertices are connected to the boundary vertices on the opposite side of the area. By $G(V, E)$, we denote the graph representing the grid (in both cases), where V is the set of vertices and E is the set of edges, respectively. We denote the number of the mobile nodes on the graph by n and the node density by $\lambda = n/|V|$.

The movement of the nodes on these two topologies differs. On a rectangular grid, the nodes that reach boundary vertices of a simulation area bounce back into the area, whereas on a torus-like grid, the nodes that reach one side of the simulation area continue travelling and reappear on the opposite side of the area.

We assume that the nodes perform a random walk on G. More precisely, the position $X_i(t)$ of a mobile node i at discrete time t is an independent random process with a stationary distribution π. This means that a node (initially located at a vertex v_0) moves at each step with an equal likelihood (1/5) to one of the neighboring vertices, or stays at its current position with the same probability (1/5). The secure side channel between two nodes can be activated when they are located on the same vertex.

We simulated the behavior of random walks for three different sizes of the rectangle grid and with two different numbers of security associations s. The results of the simulations are shown in Figure 2. The simulations were performed 20 times for each configuration and the averaged results are presented with a confidence interval of 95%.

These results show that for a small $|V|$, $t_\omega^{1\times s} \approx |V|$ and $t_\omega^{1\times s} = O(|V| \log |V|)$ in general[5]. These results can be interpreted in the following way: If a node moves according to the random walk within a given area, but in such a way that its movement is restricted to a set of meeting points connected into a grid-like structure, then the expected time at which it meets all of its desired communication partners is in the order of $|V| \log |V|$, where $|V|$ is the number of meeting points (i.e., the size of the graph).

We further observe that the convergence depends essentially on the size of the grid and not significantly on s. This means that, for a given grid size, an approximately equal fraction of the desired security associations will be established after some time t, regardless of the number of the desired security associations. This is not surprising as the nodes are uniformly distributed over the grid. Thus, the higher the density, the higher the number of security associations that a node will establish in each step. It is also worth noticing that the convergence increases very fast at the beginning, and that 90% of the desired security associations are established in approximately half of the convergence time (note that time is represented in log scale on the figures). This happens because in the first steps, a node is likely to interact with other nodes that were initially close and will, much later, meet the nodes that were initially very distant.

Here, we extend our analysis to the convergence $r^{n\times s}(t)$ and convergence time $t_\omega^{n\times s}$ of all n nodes. These values are significant as they show the time at which the system as a whole achieves the desired security properties, or more precisely, the time at which all the nodes have established the desired security associations. We simulated the time at

[5]This observation is a result of our more detailed investigation that we will publish in the future.

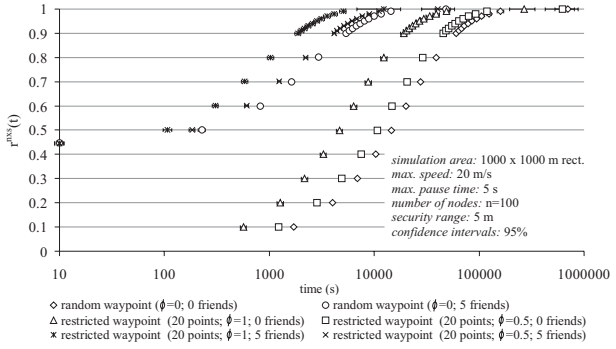

Figure 4: Convergence $r^{n\times s}(t)$ with the (restricted) random waypoint model on a 1000×1000 m rectangle for $n = 100$, $s = 99$, and various values of ϕ, with and without the friend mechanism.

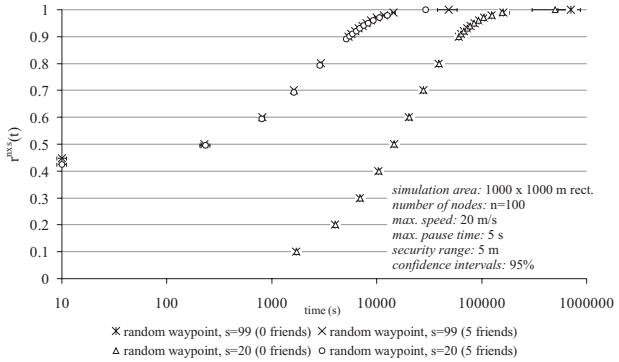

Figure 5: Convergence $r^{n\times s}(t)$ with the random waypoint model on a 1000×1000 m rectangle for $n = 100$, and for $s = 99$ and $s = 20$, with and without the friend mechanism.

which all n nodes establish security associations with all the other nodes, meaning that in matrix P, $p_{ij} = 0$ for $i = j$ and 1 otherwise. The results are shown on Figure 3. Simulations were run 20 times for each configuration and the averaged results are presented with a 95% confidence interval[6]. This figure shows that as in the single-node case, 90% of the security associations were established in approximately half of the convergence time. The figure also shows a linear (in the number of friends) decrease of the convergence time, when the friend mechanism is used. This happens for the following reason: When we extend our scheme with the friend mechanism, not only the node, but also its friends will establish security associations with the communication partners of interest. Clearly, each friend has the same chance as the node itself of meeting the node's desired communication partners, provided that the positions of the friends are initially drawn from a uniform distribution or provided that the system reached stationarity. Therefore, as confirmed by simulations, the friend mechanism should reduce the average convergence time according to the number of the friends, meaning that $t_{\omega,f} \approx t_\omega/(f+1)$, where $t_{\omega,f}$ is the average convergence time with the friends mechanism and f is the number of friends of each node.

4.3 (Restricted) random waypoint mobility

Having reached a first set of results based on the random walk, we now make use of a more realistic mobility model: random waypoint mobility model [7], which is the most commonly used mobility model for mobile ad hoc networks.

In the random waypoint mobility model, a mobile node moves on a finite continuous plane from its current position to a new location by randomly choosing its destination coordinates, its speed of movement, and the amount of time that it will pause when it reaches the destination. After the pause time, the node chooses a new destination, speed, and pause time. This is repeated for each node, until the end of simulation time.

Here, we extend this model by allowing users to move in the same way as in the random waypoint model, but their choice of destination points is restricted with some probability ϕ to a number of fixed points on a plane. This means that with probability ϕ, a node will randomly choose a point from a finite set of destination points, and with probability $1-\phi$, a node will choose as its destination any point on a plane. We call this model the *restricted random waypoint* mobility model. This model is closer to reality in the sense that users normally do not randomly choose any point on a plane as their destination, but rather they move to certain meeting points (e.g., meeting rooms, lounges, restaurants), where communication between the users takes place. If $\phi = 1$ and if the set of destination points is small, the convergence time will be very small. On the other hand, if $\phi = 0$, we have the standard random waypoint mobility model and the convergence time will be longer.

In this model, two nodes can establish a security association if they are in the *security range* of each other. The security range is significantly smaller than the power range of the mobile nodes and is the maximum range that is sufficient for the secure side channel to be set up.

Here again, like in the random walk mobility model, we have observed the creation of the security associations with two simulation areas: a bounded rectangle simulation area, where the nodes bounce off the area borders, and a torus, where nodes continue to travel through the area bounds and reappear on the other side of the area. The difference between the results in the two cases is marginal and, for the clarity of the figures, we show only the results that we obtained for the rectangular area. We note that even if the random waypoint model is continuous, we simulated it with a discrete time simulator with a time step equal to $1s$.

We observe the convergence $r^{n\times s}(t)$ and the convergence time $t_\omega^{n\times s}$ with the restricted waypoint mobility model, for different values of ϕ, with and without the friend mechanism. The simulation results are shown on Figure 4. The simulations were run 20 times for each configuration and the averaged results are presented with 95% confidence intervals.

As we expected, the results show that the convergence is much faster if the nodes choose "meeting points" as their destinations with a higher probability than any random destination points on the plane. Here, like in the random walk scenario, the convergence time remains in the order of the

[6]Note that the confidence intervals are not always visible on the figures due to the log scale.

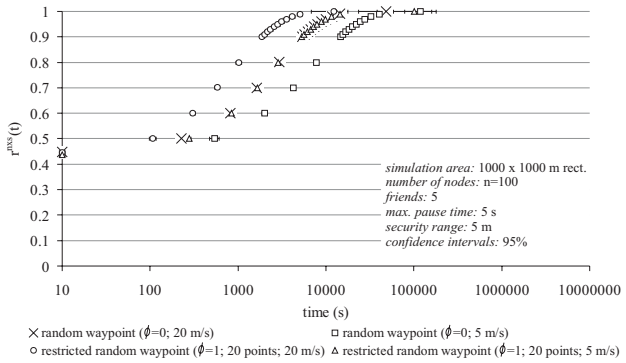

Figure 6: Convergence $r^{n \times s}(t)$ with the restricted random waypoint model on a 1000×1000 m rectangle for $n = 100$, $s = 99$, and for two different maximum speeds of node movement, with and without the friend mechanism.

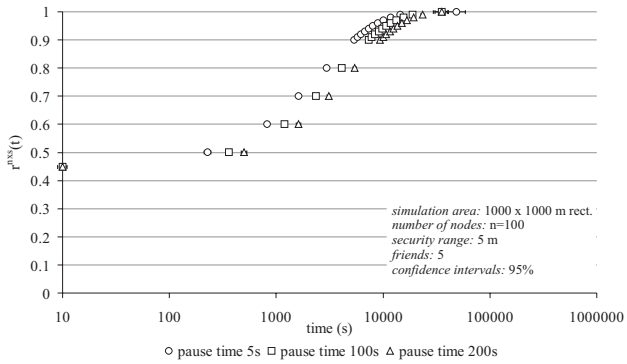

Figure 7: Convergence $r^{n \times s}(t)$ with the random waypoint model on a 1000×1000 m rectangle for $n = 100$, $s = 99$, and for several pause time durations, with the friend mechanism.

$N \log N$, where N is the size of the simulation area. Although the size of the area and time cannot be compared directly between the two scenarios, this observation is a useful indicator of the speed of creation of the security associations for a given maximum node speed, which in this case is 20 m/s. Here again, the friend mechanism speeds up the convergence and reduces the convergence time by a factor equal to the number of friends. We further simulated a scenario with $n = 100$ nodes, but where each node wants to establish security associations with $s = 20$ randomly chosen nodes on a rectangular area. These results are displayed on Figure 5.

These results show that there is no large difference in the convergence between the cases $s = 99$ and $s = 20$, but that the convergence time for the $s = 99$ case is somewhat longer.

To observe how the convergence and the convergence time vary with the speed of node movement, we observed the system behavior for two different maximum speeds (5 m/s and 20 m/s). The results of these simulations are shown on Figure 6, where simulations were run 20 times for each scenario and the averaged results are shown with a 95% confidence intervals. In compliance with our expectations, the figure shows that the convergence is faster as the speed of the node movement increases. The figure further shows that the time to establish 90% of all security associations, depending on the node speed and the presence of meeting points, ranges from a couple of hours to a couple of days.

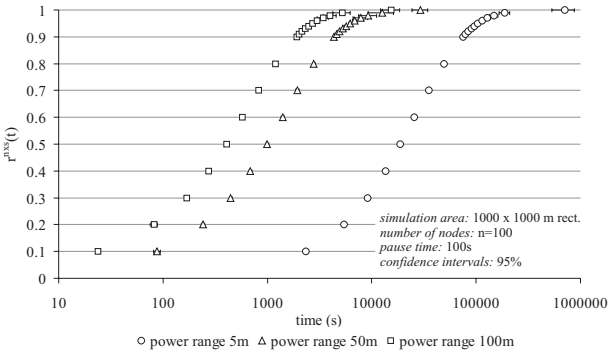

Figure 8: Convergence $r^{n \times s}(t)$ with the random waypoint model on a 1000×1000 m rectangle for $n = 100$, $s = 99$, and for several node power ranges.

Furthermore, we observed how the convergence changes with the change of pause time duration and we noticed that this change is not very significant: If the pause time is doubled, the convergence time increases by 10 to 20 percent. These results are shown on Figure 7.

The speed of establishment of security associations in authority based ad hoc networks will differ from the fully self-organized scenario because the distance at which nodes will set up security associations will be significantly increased (typically to their power range). We have simulated this by setting nodes' power ranges to 5, 50 and 100 m. The results are shown on Figure 8. As we expected, the convergence is much faster with the larger power ranges, and the convergence time is between 5000 to 10000 seconds, without the friends mechanism.

5. CONCLUSION

In this paper, we have provided a solution to the difficult problem of setting up security associations in mobile ad hoc networks. We have shown that far from being a drawback, mobility can in fact help security in such networks. We have illustrated the genericity of our approach by explaining its application to two application scenarios: fully self-organized networks and networks with an off-line authority.

For the first case, we have shown that the mechanism is highly understandable by the user, as it reproduces the concept of friends and leverages on physical encounters; these encounters make it possible for a user to associate a face to a given identity (and to a given public key), solving many of the problems of classical (hence "remote") security mechanisms (e.g., impersonation attacks and Sybil attacks [9]).

For the second case, we have shown how ad hoc networks that are secured with a central authority can also benefit from mobility. More specifically, a direct establishment of security associations over the (one-hop) radio link solves the problem known as the security-routing interdependency loop.

Simulations of our scheme in various mobility scenarios have demonstrated how the speed of establishment of security associations depends on the area size and node speed. We have observed that in the case of the random walk, where the node speed is constant, the convergence time is, in the worst case, in the order of $|V| \log |V|$, where $|V|$ is the size of the observed area; but with the friend mechanism, this time can significantly be reduced (proportionally to the number of the friends). The analysis with the restricted random waypoint model shows that for certain scenarios (100 nodes, 1 km^2, 20 m/s, 20 meeting points, 5 friends, 5 m security range), the time necessary to establish security associations between all users is around $10,000$ seconds (less than 3 hours). However, if the users do not use the friend mechanism, this time can be as long as $100,000$ seconds (around 27 hours).

These times may seem long, but we have shown that the vast majority of the security associations are set up in a much shorter time. Moreover, if the users are willing to set up security associations, they can make movement decisions in order to meet people of interest. This user-assisted approach requires short-range secure side channels to operate properly and thus the time to establish security associations can be as long as a couple of days. However, in ad hoc networks controlled by an (off-line) authority, nodes establish security associations automatically with other nodes in their power range, and convergence time becomes significantly shorter; As our simulations showed, this time is ten to twenty times shorter than in the fully self-organized scenario, typically from 5000 to 10000 seconds. It is also worth noticing that more than 70% of security associations are established within the first 1000 seconds.

Our proposed approach can also be used in other kinds of networks, notably to build up security associations at the application layer. For example, the existing Short Message Service (SMS) supported by cellular networks could be secured in a fully self-organized manner: users would initially declare who their friends are and subsequently exchange triplets by means of their infrared interfaces when they make encounters of interest to them, as described in section 3. A similar example would be to secure email between PDAs communicating with each other via wireless LANs and the Internet.

To the best of our knowledge, this paper is the first to provide a systematic study of the benefits of mobility on security; we believe that this approach can pave the way to *mobile peer-to-peer security*. This paper also shows that peer-to-peer security is *easier* to achieve in a mobile setting than in a static one; indeed, as we have seen, higher mobility leads to a faster creation of the security associations. An additional strength of our approach is that it is based exclusively on well-established cryptographic techniques.

In the future, we plan to study more realistic and more sophisticated mobility models, including those with correlated mobility patterns. We also intend to analyze the burden of the cryptographic functions on the processing units of the devices and to propose appropriate optimizations (notably by using symmetric cryptography). We further intend to investigate rekeying and key revocation schemes.

6. ACKNOWLEDGMENTS

We are thankful to Mario Čagalj and Adrian Perrig for useful comments and discussions.

7. REFERENCES

[1] N. Asokan and P. Ginzboorg. Key agreement in ad hoc networks. *Computer Communications*, 23:1627–1637, 2000.

[2] D. Balfanz, D. Smetters, P. Stewart, and H. Wong. Talking to strangers: Authentication in ad hoc wireless networks. In *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)*, 2002.

[3] N. Ben Salem, L. Buttyán, J.-P. Hubaux, and M. Jakobsson. A charging and rewarding scheme for packet forwarding in multi-hop cellular networks. In *Proceedings of the 4th ACM/SIGMOBILE MobiHoc*, 2003.

[4] L. Buttyán and J.-P. Hubaux. Enforcing service availability in mobile ad-hoc WANs. In *Proceedings of the 1st IEEE/ACM MobiHoc*, August 2000.

[5] L. Buttyán and J.-P. Hubaux. Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks. *ACM/Kluwer Mobile Networks and Applications (MONET)*, 8(5), October 2003.

[6] L. Buttyán and J.-P. Hubaux(Eds). Report on a Working Session on Security in Wireless Ad Hoc Networks. *Mobile Computing and Communications Review*, 6(4), 2002.

[7] T. Camp, J. Boleng, and V. Davies. Mobility models for ad hoc network research. *Wireless Communications and Mobile Computing (WCMC)*, Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, 2002.

[8] S. Čapkun, L. Buttyán, and J.-P. Hubaux. Self-organized public-key management for mobile ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(1), January-March 2003.

[9] J. Douceur. The Sybil attack. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[10] Henri Dubois-Ferriere, Matthias Grossglauser, and Martin Vetterli. Age matters: Efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of the 4th ACM/SIGMOBILE MobiHoc*, 2003.

[11] M. Grossglauser and D. Tse. Mobility increases the capacity of ad-hoc wireless networks. In *Proceedings of Infocom*, 2001.

[12] M. Grossglauser and M. Vetterli. Locating nodes with EASE: Mobility diffusion of last encounters in ad hoc networks. In *Proceedings of Infocom*, 2003.

[13] M. Guerrero Zapata and N. Asokan. Securing ad hoc routing protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, September 2002.

[14] J.-P. Hubaux, Th. Gross, J.-Y. Le Boudec, and M. Vetterli. Toward Self-Organized Mobile Ad Hoc Networks: The Terminodes Project. *IEEE Communications Magazine*, January 2001.

[15] Y.-C. Hu, D. B. Johnson, and A. Perrig. SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, June 2002.

[16] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. In *Proceedings of the 8th ACM MobiCom*, September 2002.

[17] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiquitous security support for mobile ad hoc networks. In *Proceedings of the 9th International Conference on Network Protocols (ICNP)*, November 2001.

[18] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of The 1st ACM MobiCom*, 2000.

[19] T. Matsumoto, Y. Takashima, and H. Imai. On seeking smart public-key distribution systems. *Transactions of the IECE (Japan)*, (69), 1986.

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

[21] G. Montenegro and C. Castelluccia. Statistically unique and cryptographically verifiable (SUCV) identifiers and addresses. In *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)*, 2002.

[22] G. O'Shea and M. Roe. Child-proof authentication for MIPv6 (CAM). *ACM Computer Communications Review*, April 2001.

[23] P. Papadimitratos and Z.J. Haas. Secure routing for mobile ad hoc networks. In *Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS)*, January 2002.

[24] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In *Proceedings of the International Conference on Network Protocols (ICNP)*, November 2002.

[25] F. Stajano. *Security for Ubiquitous Computing*. John Wiley and Sons, February 2002.

[26] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of the 7th International Workshop on Security Protocols*, 1999.

[27] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network*, 13(6):24–30, 1999.

[28] P. Zimmermann. *The Official PGP User's Guide*. MIT Press, 1995.