Mobility Helps Peer-to-Peer Security

Srdjan Čapkun, Jean-Pierre Hubaux, and Levente Buttyán
Laboratory for Computer Communications and Applications (LCA)
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

We propose a straightforward technique to provide peer-to-peer security in mobile networks. We show that far from being a hurdle, mobility can be exploited to set up security associations among users. We leverage on the temporary vicinity of users, during which appropriate cryptographic protocols are run. We illustrate the operation of the solution in two scenarios, both in the framework of mobile ad hoc networks. In the first scenario, we consider *fully self-organized* security: users authenticate each other by visual contact and by the activation of an appropriate secure side channel of their personal device; we show that the process can be fuelled by taking advantage of trusted acquaintances In the second scenario, we assume the presence of an off-line certification authority and we show how mobility helps to solve the security-routing interdependency cycle; in this case, the security protocol runs over one-hop radio links. We then show that the proposed solution is generic: it can be deployed on any mobile network and it can be implemented either with symmetric or with asymmetric cryptography. We provide a detailed performance analysis by studying the behavior of the solution on various mobility models.[1]

Index Terms: Mobile ad hoc networks, Network-level security and protection, Routing protocols

1 Introduction

Peer-to-peer security is considered to be more difficult to achieve than traditional security based on central servers. One would expect the problem to become more difficult when users are allowed to move around and to be connected only sporadically. Indeed, according to common belief, wireless communication and mobility are at odds with security: jamming or eavesdropping is easier on a wireless link than on a wired one, notably because such mischief can be perpetrated without physical access or contact; likewise, a mobile device is more vulnerable to impersonation and to denial of service attacks.

The security architectures of existing mobile networks are highly centralized (as are their static, wireline counterparts). For example, the security of GSM relies on a key, shared by the subscriber and the operator, which is established at the time the contract is signed; the security of Third Generation Cellular Networks is based on the same principle. Another example is the *Wireless Transport Layer Security* (WTLS) protocol, aimed at providing secure Web access from a mobile device: the servers are authenticated by a certificate of their public key, delivered by a well-established certification authority.

Both examples show that the driving (and understandably so) security concern has been to serve the interest of specific organizations: In the first case, the security system guarantees an operator that only legitimate subscribers can make use of the communication service it provides; in the second case, it lets an e-business company claim to its own customers that they are connected to the right Web server and that the message exchange is protected.

So far, nothing has been proposed for *peer-to-peer* security in mobile networks. We will show that, far from being a hurdle, mobility can in fact help security by enabling basic functions such as authentication and key establishment; we will even demonstrate that the higher the speed of the nodes, the higher the pace at which the security associations will be established. We will illustrate the principles of our solution in two scenarios, both in the area of mobile ad hoc networks.

In the first scenario, 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; when they are close to each other, users can activate a *secure side channel* between their personal devices to authenticate each other and set up shared keys.

[1] EPFL-IC Technical report no. IC/2003/81

The second scenario corresponds to situations where an (off-line) authority provides the authorization to each mobile node to join the network, but it does so only at the initialization of each node; when in each others' radio range, nodes mutually authenticate and set up shared keys; this approach allows the nodes to join the network at different times. An important use of this approach is to secure routing as the direct (one-hop) establishment of security associations breaks the routing-security interdependence cycle [26].

As we will argue in the conclusion, the proposed solution can be applied in any mobile network, including cellular networks.

In [30], we have developed this initial idea by quantifying the benefits of mobility and by introducing a novel mechanism ("friends") that supports the establishment of security associations even between nodes that do not meet physically. Here, we further extend this work by proposing protocols that allow the implementation of our system with both symmetric and public-key cryptography. Moreover, to better observe the establishment of security associations, we present analytical results and extend our simulation results with additional mobility models.

The work presented in this paper is part of the Terminodes Project [3, 23].

The organization of the paper is the following. In Section 2, we survey the related work. In Section 3, we describe our system model, as well as the two scenarios mentioned above. In Section 4, we explain how security associations are created based on encounters and we provide the cryptographic protocols. In Section 5, we study in detail the pace at which the security associations are created, by taking several mobility models into account. Finally, we conclude the paper in Section 6.

2 STATE OF THE ART

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

In [32], 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, that is used to verify/sign public-key certificates of the network nodes. The private key k 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. Signatures are then generated by a collaborative action of the servers. 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. 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. [13] describes a similar approach, that 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, and an 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. A disadvantage is that the first $t+1$ nodes must be initialized by a trusted authority; it is also unclear how the value of t can be changed in case the overall number of nodes significantly increases (or decreases). Furthermore, the system seems to be vulnerable to the Sybil attack [8]: 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*. The proposed solution is the following: A fresh password is chosen and shared among users (e.g., it is written on a blackboard). To prevent *dictionary attacks* [16], this password is not used directly; instead, 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 approach has the drawback of being somewhat cumbersome, as it requires the users to type the password in their personal device and to be present in the same room.

Another approach, originally designed for the address ownership problem in Mobile IPv6, is described by Montenegro and Castelluccia in [17] and by O'Shea and Roe in [18]. 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 a need for certificates that bind the node's address to its public key, because 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 [29], 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. 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 the local detection of inconsistent certificates and the use of authentication metrics.

Finally, we must mention the works of Grossglauser and Tse [11], and Grossglauser and Vetterli [12]; these papers show that mobility can help to increase the per-user throughput in ad hoc networks and to disseminate destination location information without incurring any communication overhead to the network. A more recent work by Dubois-Ferriere, Grossglauser and Vetterli [9] 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 [28, 15, 27, 22, 19, 5].

3 System Model

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

3.1 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 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 u can relate the name (or the face) of another user v to his (v's) public key, we will say that there is a *one-way security association* from u to v. Two one-way security associations between u and v (one in each direction) constitute a two-way security association between u and v; in the rest of the paper, security associations will be considered to be two-way if not mentioned otherwise. Moreover, if a user u can relate the name (or the face) of another user v with their shared secret key, we also say that there exists a two-way security association between u and v.

When they meet, users are obviously 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 almost 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 an identificator (e.g., 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

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

wants to establish a security association with a user-independent device (e.g., a printer), she will visually identify the device 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.

These encounters make it possible for a user to associate a face to a given identity (and to a given public key), thus solving many of the classical problems of security in distributed systems (e.g., impersonation attacks and Sybil attacks [8]).

As we will show in more detail in Section 4, our system can be implemented either with public key or with symmetric key cryptography.

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.

To expedite the process of establishment of security associations, we assume that nodes can also rely on *friends*. Two nodes u and v 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. 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.

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 for a long time.

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 [29]).

3.2 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.

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

When two nodes move into the 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 well-known routing-security interdependence cycle [26]: 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 nodes 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 [26]. 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 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; a drawback is that the establishment of the security associations requires some time, as detailed in Section 5.

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 would not have a 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, whereas in the authority-based approach, users do not need to be aware of the establishment of the security associations, as it is done automatically by their nodes. The use of either of these approaches strongly depends on the purpose of the network. Typically, the self-organized approach is useful in securing personal communications on the application level, whereas the authority-based approach is used to secure networking mechanisms such as routing.

4 MECHANISMS TO ESTABLISH SECURITY ASSOCIATIONS

4.1 Public-key based approach

4.1.1 Preliminaries

This approach is based on the assumption that each node has a public key and a corresponding private key. In fact, in order to support different applications, nodes may have several public/private key pairs. However, for the sake of simplicity, in the rest of the paper, we will assume that each node has a single public/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. In the fully self-organized model, the key pair of a node is generated by the node itself. In the authority-based model, the key pair of a node can be generated by the node or by the authority; in both cases, the authority issues a certificate that binds the public key of the node to the name of the user that is associated with the node.

Each node also has a node address that is used by the routing protocol. In the fully self-organized model, the node address is generated from the public key of the node[3] by making use of a technique similar to CAM [18] or SUCV [17]. In this way, node addresses are bound to public keys in a verifiable way. Note, however, that a malicious node may generate several public keys and corresponding 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. In the authority-based model, the node address is bound to the public key of the node by the certificate issued by the authority; this removes the need for CAM, SUCV, or similar mechanisms.

4.1.2 Security associations

A (two-way) security association between two nodes u and v is represented by a triplet $(U,\ k_u,\ a_u)$ at the side of v and a triplet $(V,\ k_v,\ a_v)$ at the side of u, where U and V are the names of the users that are associated with nodes u and v; k_u and k_v are the public keys of u and v[4]; and a_u and a_v are the node addresses of u and v, respectively.

Once a security association between two nodes u and v is established, they can set up secure communication channels that protect the integrity and confidentiality of the exchanged messages. In fact, for efficiency reasons, u and v may want to use symmetric key cryptography for the protection of their messages; in this case, they establish short-term symmetric keys (session keys) using the public keys in the security association.

In the public-key based approach, it is possible to establish a one-way security association between two nodes u and v. This occurs if only one of the nodes, say u, has the triplet $(V,\ k_v,\ a_v)$. In this case, u can check the node address of v, verify messages signed by v, and send encrypted messages to v, but not vice versa. 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 between them; therefore, security associations can be established in two steps by setting up a one-way security association in each step.

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

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

4.1.3 Establishment of security associations

In the authority-based model, two nodes can establish a security association by exchanging their certificates. This is rather straightforward and we do not detail it further. Instead, we focus on the establishment of security associations in the fully self-organized model. Here, three mechanisms support the establishment of new security associations (Figure 1).

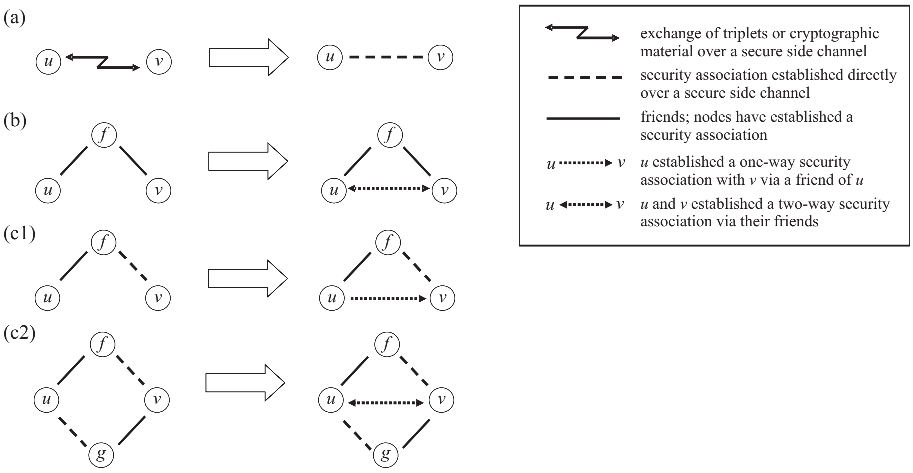

(a)
u
v
u
v
(b)
f
u
v
f
u
v
(c1)
f
u
v
f
u
v
(c2)
f
u
v
g
f
u
v
g
exchange of triplets or cryptographic material over a secure side channel
security association established directly over a secure side channel
friends; nodes have established a security association
u v
u established a one-way security association with v via a friend of u
u v
u and v established a two-way security association via their friends

Figure 1. Three mechanisms to create new security associations using (a) the secure side channel, (b) a common friend, and (c1,c2) the combination of the first two approaches ((c1) mechanism is used only in the public-key based approach).

The first mechanism is used when two nodes u and v are in the vicinity of each other, and it consists in u and v exchanging 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. This guarantees a secure binding between the received user name, public key, and node address. In addition, the user can easily verify the validity of the received name because the name should correspond to the person present at the encounter. The 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 address can be verified against the public key.

Protocol 1 shows a possible implementation of this mechanism.

Protocol 1: Direct Establishment of a Security Association

msg1 (secure side ch.)	$u \rightarrow v:$	$a_u \mid \xi_u = h(r_u \mid U \mid k_u \mid a_u)$
msg2 (secure side ch.)	$v \rightarrow u:$	$a_v \mid \xi_v = h(r_v \mid V \mid k_v \mid a_v)$
msg3 (radio ch.)	$u \rightarrow v:$	$r_u \mid U \mid k_u \mid a_u$
msg4 (radio ch.)	$v \rightarrow u:$	$r_v \mid V \mid k_v \mid a_v$
	$u:$	$h(r_v \mid V \mid k_v \mid a_v) = \xi_v?;\ V?;\ match(k_v, a_v)?$
	$v:$	$h(r_u \mid U \mid k_u \mid a_u) = \xi_u?;\ U?;\ match(k_u, a_u)?$
msg5 (radio ch.)	$u \rightarrow v:$	$\sigma_u(r_v \mid U \mid V)$
msg6 (radio ch.)	$v \rightarrow u:$	$\sigma_v(r_u \mid V \mid U)$

In Protocol 1, u and v first generate random numbers r_u and r_v, respectively, and exchange, through the secure side channel, their addresses a_u and a_v and the cryptographic hash values $\xi_u = h(r_u \mid U \mid k_u \mid a_u)$ and $\xi_v = h(r_v \mid V \mid k_v \mid a_v)$ of their random numbers and triplets. After this initial exchange, u and v send messages to each other through the radio interface (since they have obtained each other's node address in the first two messages). They exchange their random numbers and triplets, and each of them verifies if the hash value of the received random number and triplet is equal to the received hash value ξ_u (or ξ_v). 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. The random numbers serve as nonces and guarantee the freshness of the messages that follow. Now, both users can verify if the received user name corresponds to the other party and both nodes can verify if the received node address matches the received public key. Finally, the nodes generate and send to each other a signature ($\sigma()$) on the received random number and the user names in order to prove that they possess the private keys that belong to the exchanged public keys.

With the second mechanism, two nodes u and v can establish a security association if they have a common friend f. A simple solution is the following: Since f knows the triplets of both u and v, it can issue (on request from u and/or v) fresh certificates for both triplets and send them to v and u, respectively, via the network. Both u and v 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 is a combination of the friendship relationships and the encounters, and it establishes only a one-way security association: If nodes u and f are friends and f has obtained the triplet of v in an encounter with v, then f can issue (on request from u) a fresh certificate for the triplet of v, and send this certificate to u via the network. Since u 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. A two-way security association between nodes u and v is then established as a combination of two one-way security associations (from u to v and from v to u).

The protocols corresponding to the second and third mechanisms are straightforward and we do not detail them.

4.2 Symmetric-key based approach

The authority-based model is inherently public-key based, but the fully self-organized model can be based on symmetric-key instead of public-key cryptography. The immediate advantage of using symmetric-key cryptography is the reduced computational overhead of the mobile nodes.

In the symmetric-key based approach, the nodes do not have any long-term keys (such as the key pairs in the public-key based approach). Instead, each node sets up session keys with the nodes with which it wants to communicate. Node addresses therefore cannot be computed from long-term keys; they are computed from user names in a cryptographically verifiable manner (e.g., the user name is extended with some random string and hashed).

4.2.1 Security associations

A security association between nodes u and v is represented by a triplet $(U,\ k_{uv},\ a_u)$ at the side of v and a triplet $(V,\ k_{uv},\ a_v)$ at the side of u, where U and V are the names of the users that are associated with nodes u and v; k_{uv} is a symmetric key shared by u and v; and a_u and a_v are the node addresses of u and v, respectively. Once a security association is established between two nodes u and v, they can communicate securely in both directions (from u to v as well as from v to u) due to the symmetric nature of the key k_{uv}. Therefore, in the symmetric-key based approach, we consider security associations to be always two-way; it is not possible to establish a one-way security association[5].

4.2.2 Establishment of security associations

Here as well, there are three mechanisms to establish security associations. The first one is the direct establishment through the side channel: When the nodes are in the vicinity of each other, they can exchange, through the side channel, their user names and node addresses, and additional data that allow them to compute a shared secret. It is important to note, however, that in a pure symmetric-key approach, setting up a shared secret between two parties always requires a confidential channel between them. This means that in this case, the side channel must ensure not only the integrity but also the confidentiality of messages. Like in the public-key implementation, the users can verify the received names through personal encounters. The node addresses, on the other hand, can be verified against the received (and verified) names.

The second mechanism supports the establishment of security associations between two nodes u and v via a common friend f. By assumption, f already has a security association with both u and v, meaning that it has symmetric keys established with them. In addition, f is trusted by both u and v. Therefore, to establish a session key between u and v, well-known symmetric-key protocols can be used, where f plays the role of the trusted (key)

[5] In practice, the nodes may derive sub-keys from the shared symmetric key of the security association, where each sub-key is used in one direction only and perhaps only for a specific security service (e.g., either for integrity or for confidentiality, but not for both), but this is a policy issue, out of the scope of our discussion.

server. The session key can be generated either by f who would send it to both u and v (like in the Kerberos protocol), or by u or v, in which case f would be used as a trusted relay (like in the Wide-Mouth-Frog protocol [4]).

Finally, the third mechanism can be used when two nodes u and v do not have a common friend, or have a common friend f but do not want f to know their shared secret key. Like the third mechanism in the public-key based approach, this mechanism combines the first two mechanisms (encounters and friends). Let us assume that u has a friend f who has already set up a security association with v using the first mechanism. Similarly, let us assume that v has a friend g who has set up a security association with u using the first mechanism. Now u and v can set up a security association using f and g by u generating key contribution k_u and sending it to v via g, v generating key contribution k_v and sending it to u via f, and then both u and v computing a common value k_{uv} from k_u and k_v. Protocol 2 illustrates this in more detail. In this protocol, nodes u and v first exchange the names of their friends to be used in the protocol as trusted relays, and two nonces r_u and r_v that are used to guarantee the freshness of subsequent messages. Then, u generates some random key k_u and sends it to v via g (msg3 and msg4), and v generates some random key k_v and sends it to u via f (msg3' and msg4'). Here, $d_{x \to y}$ is a direction bit that indicates that the message goes from x to y (and not from y to x)[6]. req and rep are bits that indicate that the message is a *request* to a friend or a *reply* from a friend, respectively. We need these bits because every node can play either the role of a requesting node (u and v) or the role of a friend (f and g), and thus we must indicate not only that this is a message from x to y but also that x is the requesting node and y is the friend (or vice versa). Finally, u and v compute a common value k from k_u and k_v using a publicly known pseudo-random function h (e.g., a hash function).

Protocol 2: Friend-Assisted Establishment of a Security Association

msg1	$u \to v:$	f, r_u
msg2	$v \to u:$	g, r_v
msg3	$u \to g:$	$u, \{d_{u \to g}, req, v, k_u, r_v\}k_{ug}$
msg4	$g \to v:$	$g, \{d_{g \to v}, rep, u, k_u, r_v\}k_{vg}$
msg3'	$v \to f:$	$v, \{d_{v \to f}, req, u, k_v, r_u\}k_{vf}$
msg4'	$f \to u:$	$f, \{d_{f \to u}, rep, v, k_v, r_u\}k_{uf}$
	$u, v:$	$k_{uv} = h(k_u, k_v)$

An interesting feature of the protocol is that it replaces a single trusted party with two parties trusted by one entity each. If f and g are not colluding, then none of them have enough information to compute k_{uv}. In addition, both u and v trust at least one of them for not colluding.

4.3 Brief comparison of the two approaches

The advantages of the public-key based approach are the following: 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. Second, friends can distribute public keys (e.g., in form of certificates) without being able to decrypt messages that are encrypted with those public keys or to forge signatures that can successfully be verified with those public keys. In the symmetric-key based approach, the side channel must provide protection against eavesdropping and colluding friends can learn information that can be misused.

The symmetric-key based approach reduces the computational burden on the nodes, since they do not have to perform expensive public key cryptographic operations at all. In addition, long-term secrets must not be stored by the nodes, which makes key revocation much less of a problem. Moreover, compromise of a symmetric key affects only one security association, whereas compromise of the private key of a node requires *all* of its security associations to be reset.

4.4 Key Revocation and Rekeying

Having presented the protocols for the establishment of security associations, we now discuss how key revocation and rekeying can be performed in our system.

In fully self organized ad hoc networks, keys (public or symmetric) are bound to users' names. If a user's key is compromised, she sends a key revocation message, signed (or encrypted) with the compromised key to all users

[6] Note that since messages are always encrypted with a symmetric key k_{xy}, the only ambiguity could be the direction.

who made use of this key to forge a security association with her. This can be done in the following ways: (i) by multicasting the revocation message to these users, (ii) by performing the revocation when she encounters these users or (iii) by performing a limited (in number of hops) multicast periodically, until all users get the revocation message. These solutions differ in speed and communication cost of revocation.

In the authority based ad hoc networks, a node address is bound to its public key by a certificate issued by the central authority. In this scenario, either the authority or the owner of the device can perform revocation, and they can do so for different reasons: the user can revoke her node's public key if she believes that the key is compromised, whereas the authority can revoke the node's (user's) key for various reasons (e.g., to enforce some network policy). If the user revokes the node's key, the mechanisms that she can use are the same as in fully self-organized networks. On the other hand, key revocation by the central authority can be very different, especially if we assume the authority to be off-line. In this case, the authority can issue certificates of a limited time validity, inject a revocation certificate directly into a network, or revoke the node's key when introducing a new node in the network.

An additional interesting low-cost mechanism, enabled by the mobility based scheme, is rekeying. This simple, supplementary mechanism enables nodes to renew their security associations whenever they meet and thus reduces the chance that, even if a key is compromised, the attacker will be able to use it. In Section 5 we study the rekeying frequency with different mobility models.

In the future, we intend to study key revocation and rekeying mechanisms in more details.

5 Performance Evaluation

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

5.1 Terminology

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

More specifically, matrix $F = [f_{uv}]$ is defined as follows: $f_{uv} = 1$ if u trusts v (i.e., v is a friend of u); 0 otherwise. Matrix $P = [p_{uv}]$ captures the fact that in the general case, a given user is interested in communicating with only a subset of the other users: $p_{uv} = 1$ if u wants to establish a one-way security association with v; 0 otherwise. Finally, $E(t) = [e_{uv}(t)]$ represents the evolution of the security associations over time: $e_{uv}(t) = 1$ if at time t, u established a one-way security association with v; 0 otherwise. Matrix F is symmetric, whereas matrices P and $E(t)$ are generally not.

Regardless of the kind of the cryptosystem used, we should note that it is possible to exploit friendships even before the nodes start moving: if $f_{ug} = 1$ and $f_{gv} = 1$, then $e_{uv}(t_0)$ and $e_{vu}(t_0)$ 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 different mobility models.

In our analysis, we will observe the following values: the convergence $r(t)$, which represents the fraction of the required security associations established until time t, and the convergence[7] (meeting) time t_M, which is the time needed to establish all the desired security associations. Thus $r(t)$ is computed as

$$r(t) = \frac{\sum_{u,v} e_{uv}(t) \cdot p_{uv}}{\sum_{u,v} p_{uv}} \qquad (1)$$

and the convergence time t_M is the earliest time at which $r(t) = 1$.

[7] We use the terms meeting time and convergence time interchangeably.

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

One additional value of interest is the average inter-meeting time t_{IM} of nodes, which shows how frequently the nodes meet. This value is important for assessing the frequency of rekeying and the time necessary to perform key revocation.

In our analysis, we will make use of the following mobility models: Random walk, Random Waypoint [7], Restricted Random Waypoint [30], Gauss-Markov [14], and Modified Gauss-Markov. For the random walk mobility model, due to its simplicity, we provide closed form analysis. For the other mobility models, we rely primarily on simulations.

5.2 Random walk analysis

Although, in reality, node positions are continuous processes in continuous time, in this model we use a discrete approximation and we assume that mobile nodes perform simple independent random walks on a rectangular grid topology. We reckon that a discrete random walk is not an accurate representation of the movement of people or vehicles. However, the simplicity of this model allows us to obtain closed form results for the quantities of interest. In the random walk model, two nodes can activate a secure side channel between them when they are located on the same vertex.

As already mentioned, we assume that nodes perform simple, independent random walks on a rectangular toroidal grid. More precisely, we assume that the grid has continuous boundary conditions, meaning that its boundary vertices are connected to the boundary vertices on the opposite side of the grid. We have chosen to work with a toroidal grid to simplify our computations, but as we will show by simulations, all the results can be equally applied (some approximatively) to grids with reflecting boundaries.

By $G(V,E)$, we denote the graph (grid), where V is the set of vertices $(|V| = N)$ and E is the set of edges $(|E| = m)$, respectively. We assume G to be undirected and connected.

We represent the node movement on the grid as a Markov chain. The state space $\mathbb{S}$ of this chain consists of the vertices of G scanned in any order to form a vector of length N. A node (initially located at a state (vertex) $i \in \mathbb{S}$) moves at each step with an equal probability ($\frac{1}{4}$) to one of the neighboring states. More precisely, the position $X_u(t)$ of a mobile user u at discrete time t is an independent Markov process with a stationary distribution $\pi = \{\pi_i : i \in \mathbb{S}\},\ \forall i \in \mathbb{S}$; for a grid, $\pi_i = \frac{1}{N}, \forall i \in \mathbb{S}$. The transition probabilities of this chain are therefore given by:

$$p_{ij} = \begin{cases} 1/4 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $p_{ij} = P(X_u(t+1) = j | X_u(t) = i)\ \forall i,j \in \mathbb{S}$.
Our objects of study are meeting times of nodes on G. More specifically, we will observe the average and worst case mean meeting times, their distribution, and the inter-meeting times of nodes. We will further observe how these values change with the increase in the number of node's friends (f), communication partners (s) and size of the grid (N). For most of our results, we assume that nodes start their movement from a stationary distribution[8].

5.2.1 Average and worst case mean meeting times

The *average mean meeting time* $\overline{t_M}$ of two nodes u and v is the average, over all pairs of vertices, of the expected meeting times of two nodes, given that u and v start their walks at these pairs of vertices. Thus,

$$\overline{t_M} \quad = \quad \frac{1}{N^2} \sum_{i,j \in \mathbb{S}} E[t_M | X_u(0) = i, X_v(0) = j] \tag{3}$$

where $t_M = \min\{t : X_u(t) = X_v(t)\}$ is the (first) meeting time of u and v.

[8] Here, we assume, without any lose of generality, that the distances between initial positions of the nodes are even, which ensures that the nodes eventually meet.

The *worst case mean meeting time* τ_M is defined as the maximum, over all pairs of initial states of u and v, of the expected meeting time of u and v. Thus,

$$\tau_M \quad = \quad \max_{i,j \in \mathbb{S}} E[t_M | X_u(0) = i, X_v(0) = j] \tag{4}$$

Result 1: $\overline{t_M}$ and τ_M for two nodes. Given two simple, symmetric random walks on a grid, their average and worst case expected meeting times are given by

$$\overline{t_M} \approx 0.17N \log N; \qquad \tau_M \approx 0.183N \log N \tag{5}$$

Proof: We make use of the fact that the u's and v's joint Markov chain $(X_u - X_v)(t)$ behaves precisely as $X_u(2t)$, a single random walk with transition rates doubled. Hence, the expected meeting time of u and v, given that the nodes start from vertices i and j, respectively, is exactly half of the expected hitting time (t_H) of a single node, starting from i to hit j [24]. From this, it simply follows that $\overline{t_M} = \frac{1}{2}\overline{t_H}$ and $\tau_M = \frac{1}{2}\tau_H$, where $\overline{t_H}$ and τ_H are the worst case and the average expected hitting time, respectively. From [10], we have that $\overline{t_H} \approx 0.34N \log N$ and $\tau^* \approx 0.73N \log N$ as $N \to \infty$, where τ^* is the maximum mean commute time. This approximation is valid already for $N \geq 25$ [10]. Due to the torus symmetry, we have that $\tau_H = \frac{1}{2}\tau^*$. Thus, $\overline{t_M} \approx 0.17N \log N$ and $\tau_M \approx 0.183N \log N$. □

Result 2: $\overline{t_M}$ and τ_M for two nodes with friends. We extend our analysis to the friends mechanism. We observe the first time at which node u or one of its f friends meet node v. The average and the worst case of this time is given by

$$\overline{t_M}^{(f)} \approx \frac{0.17N \log N}{f+1}; \qquad \tau_M^{(f)} \approx \frac{0.183N \log N}{f+1} \tag{6}$$

Proof: We use a very intuitive argument: the Markov chain $\min\{(X_u - X_v)(t), (X_{u1} - X_v)(t), ..., (X_{uf} - X_v)(t)\}$ of u, v and u's friends runs at a $(f+1)$-times faster rate than $(X_u - X_v)(t)$. □

5.2.2 Meeting time distributions

The hitting time distribution of an ergodic Markov chain can be approximated by an exponential distribution of the average mean hitting time of the same chain [21, 25, 24]. For the meeting time, we can directly apply the same approximation. Thus, the cumulative distribution function (cdf) of the meeting time of two nodes is given by

$$P(t_M \leq t) \quad \approx \quad 1 - e^{\frac{-t}{kN \log N}} \tag{7}$$

where $k = 0.17$. Note that this is a continuous time representation of the discrete Markov chain, but the result equally applies to the discrete time case.

Result 3: Meeting time cdf for two nodes with friends. The cdf of the first meeting time of a node u or its friends with another node v is given by

$$P(t_M^{(f)} \leq t) \quad \approx \quad 1 - e^{\frac{-t}{k\frac{N}{f+1}\log N}} \tag{8}$$

Proof:

$$P(t_M^{(f)} \leq t) \quad = \quad 1 - [P(t_M > t)]^{f+1} \approx 1 - e^{\frac{-t}{k\frac{N}{f+1}\log N}}$$

□

Result 4: Meeting time cdf for $1 \times s$ nodes with friends. By $t_M^{(f),1\times s}$ we denote the first time at which node u, with the help of its friends, meets all other s nodes. The distribution of $t_M^{(f),1\times s}$ is given by

$$P(t_M^{(f),1\times s} \leq t) \quad \approx \quad [1 - e^{\frac{-t}{k\frac{N}{f+1}\log N}}]^s \tag{9}$$

Proof: The event that node u or its friends meet node v is independent of the event that they meet some other node. Thus, we can write

$$P(t_M^{(f),1\times s} \leq t) \quad = \quad [P(t_M^{(f)} \leq t)]^s \approx [1 - e^{\frac{-t}{k\frac{N}{f+1}\log N}}]^s$$

□

Result 4': Expected meeting time $\overline{t_M}^{(f),1\times s}$ for $1 \times s$ nodes with friends.

$$\overline{t_M}^{(f),1\times s} = \overline{t_M}^{(f)} \sum_{\ell=1}^{s} \frac{1}{\ell} \tag{10}$$

Proof:

$$\begin{aligned}
\overline{t_M}^{(f),1\times s} &= E[\max_{v\in\mathbb{P}_u} t_M^{(f)}(u,v)] \qquad (11)\\
&= \int_0^\infty [1 - P(t_M^{(f),1\times s} \le t)]dt \;=\; \int_0^\infty [1-[1-e^{-\lambda t}]^s]dt\\
&= \int_0^\infty [1 - \sum_{\ell=0}^{s} \binom{s}{\ell}(-1)^\ell e^{-\lambda \ell t}]dt \;=\; \sum_{\ell=1}^{s}[\binom{s}{\ell}(-1)^{\ell+1}\frac{1}{\lambda\ell}]\\
&= \frac{1}{\lambda}\sum_{\ell=1}^{s}\frac{1}{\ell} \;=\; \overline{t_M}^{(f)} \sum_{\ell=1}^{s}\frac{1}{\ell} \qquad (12)
\end{aligned}$$

where $\mathbb{P}_u$ is the set of u's communication partners ($|\mathbb{P}_u| = s$) and $\lambda = \frac{1}{k\frac{N}{f+1}\log N}$. □

Result 5: Cdf of the time $t_M^{(f),1\times s}(r')$ at which convergence $r \ge r'$ for $1 \times s$ nodes with friends.

$$P(t_M^{(f),1\times s}(r) \le t) = \sum_{i=\lceil r's \rceil}^{s} \binom{s}{i}[1-e^{-\lambda t}]^i[e^{-\lambda t}]^{s-i}$$

Proof:

$$P(t_M^{(f),1\times s}(r') \le t) = \sum_{i=\lceil r's \rceil}^{s} \binom{s}{i}[P(t_M^{(f)} \le t)]^i[P(t_M^{(f)} > t)]^{s-i} = \sum_{i=\lceil r's \rceil}^{s} \binom{s}{i}[1-e^{-\lambda t}]^i[e^{-\lambda t}]^{s-i}$$

□

Result 5': Average expected meeting time $\overline{t_M}^{(f),1\times s}(r)$ at which convergence $r \ge r', r' < 1$ for $1 \times s$ nodes with friends.

$$\begin{aligned}
\overline{t_M}^{(f),1\times s}(r') &= \frac{1}{\lambda}[\sum_{\ell=1}^{s}\frac{1}{\ell} - \sum_{\ell=\lceil r's \rceil}^{s-1}\frac{1}{s-\ell}] \qquad (13)\\
&= \overline{t_M}^{(f),1\times s} - \frac{1}{\lambda}\sum_{\ell=\lceil r's \rceil}^{s-1}\frac{1}{s-\ell} \qquad (14)
\end{aligned}$$

Proof:

$$\begin{aligned}
\overline{t_M}^{(f),1\times s}(r) &= \int_0^\infty [\sum_{i=\lceil r's\rceil}^{s}\binom{s}{i}[1-e^{-\lambda t}]^i[e^{-\lambda t}]^{s-i}]dt & (15)\\
&= \int_0^\infty [1-(1-e^{-\lambda t})^s - \sum_{i=\lceil r's\rceil}^{s-1}\binom{s}{i}[1-e^{-\lambda t}]^i[e^{-\lambda t}]^{s-i}]dt\\
&= \frac{1}{\lambda}\sum_{\ell=1}^{s}\frac{1}{\ell} - \int_0^\infty [\sum_{i=\lceil r's\rceil}^{s-1}\binom{s}{i}\sum_{j=0}^{i}\binom{i}{j}(-1)^j e^{-\lambda jt}e^{-\lambda(s-i)t}]dt\\
&= \frac{1}{\lambda}\sum_{\ell=1}^{s}\frac{1}{\ell} - \sum_{i=\lceil r's\rceil}^{s-1}\binom{s}{i}\sum_{j=0}^{i}\binom{i}{j}(-1)^j\frac{1}{\lambda(s-i+j)}\\
&= \frac{1}{\lambda}\sum_{\ell=1}^{s}\frac{1}{\ell} - \sum_{i=\lceil r's\rceil}^{s-1}\binom{s}{i}\frac{i!(s-i-1)!}{\lambda s!}\\
&= \frac{1}{\lambda}[\sum_{\ell=1}^{s}\frac{1}{\ell} - \sum_{\ell=\lceil r's\rceil}^{s-1}\frac{1}{s-\ell}] & (16)
\end{aligned}$$

□

Result 6: Meeting times for $n \times s$ nodes with friends. These values can be bounded by

$$\begin{aligned}
\overline{t_M}^{(f),n\times s} &< \overline{t_M}^{(f),1\times ns} & (17)\\
\overline{t_M}^{(f),n\times s}(r') &< \overline{t_M}^{(f),1\times ns}(r') & (18)
\end{aligned}$$

Proof: Given that the meeting event of a node u with a node v is not independent of the meeting event of a node w with nodes u and v, the distribution of $t_M^{(f),n\times s}$ can be lower-bounded by the distribution of $t_M^{(f),1\times ns}$, the distribution of the time at which a single node meets ns other nodes. Thus, $\overline{t_M}^{(f),n\times s}$ can be upper-bounded by $\overline{t_M}^{(f),1\times ns}$. The same argument is valid for $\overline{t_M}^{(f),n\times s}(r')$. □

5.2.3 Inter-meeting times

Inter-meeting time t_{IM} of two nodes u and v is the time at which two nodes meet, given that they start their walk from the same state (vertex). Thus,

$$t_{IM} = \min\{t > 0 : X_u(t) = X_v(t) | X_u(0) = X_v(0)\} \tag{19}$$

Result 7: Inter-meeting time of two nodes. The expected inter-meeting time $E[t_{IM}]$ of two nodes performing a random walk on a grid is given by [21, 24]

$$E[t_{IM}] = \frac{N}{2} \tag{20}$$

Result 8: Inter-Meeting time of two nodes with friends The expected inter-meeting time of a single node with another node or its friends is given by,

$$E[t_{IM}^{(f)}] = \frac{N}{2(f+1)} \tag{21}$$

Proof: We represent the movement of nodes on G by a Markov chain $\min\{(X_u - X_v)(t), (X_{u1} - X_v)(t), ..., (X_{uf} - X_v)(t)\}$, where $X_{u1}, ..., X_{uf}$ are positions of u's friends at time t. The state space of this chain is then $\mathbb{S}' = \mathbb{S}^{f+2}$, and its stationary distribution $\rho = \{\rho_i : i \in \mathbb{S}^{f+2}\}$. We denote by $\mathbb{M} \subseteq \mathbb{S}'$ the set of states of that chain in which node v meets node u or one of u's friends. Then, the stationary distribution of the set $\mathbb{M}$ can be calculated as $\rho(\mathbb{M}) = \frac{|\mathbb{M}|}{|\mathbb{S}'|}$. By Kac's formula [21] we have:

$$\begin{aligned} E[t_{IM}^{(f)}] &= \frac{1}{\rho'(\mathbb{M})} = \frac{|\mathbb{S}'|}{|\mathbb{M}|} && (22)\\ &= \frac{N(\frac{N}{2})^{f+1}}{N(f+1)(\frac{N}{2})^{f}} = \frac{N}{2(f+1)} \\ &= \frac{E[t_{IM}]}{f+1} && (23) \end{aligned}$$

□

5.3 Simulation setting

Having reached a first set of results based on the Random Walk, we now make use of more realistic mobility models: Random Waypoint, Restricted Random Waypoint, Gauss-Markov and Modified Gauss-Markov. First, we describe the models.

5.3.1 Mobility models

The **Random Waypoint** mobility model [20, 7] is the most commonly used mobility model for mobile ad hoc networks. In this 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.

The **Restricted Random Waypoint** model is a modification of the Random Waypoint model, where we expect nodes to move in the same way as in the Random Waypoint model, but we restrict their choice of destination points with some probability ϕ to a number of fixed points on the plane. This means that with probability ϕ, a node randomly chooses a point from a finite set of destination points, and with probability $1 - \phi$, it chooses as its destination any point on the 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 they rather move to some meeting points (e.g., meeting rooms, lounges, restaurants) where communication between users takes place. If $\phi = 1$ and if the set of destination points is small, the convergence time will be very small. On the contrary, if $\phi = 0$, we have the standard Random Waypoint mobility model and the convergence time will be longer.

The **Gauss-Markov** mobility model [14] was developed to represent different levels of node motion randomness via a single parameter. In this model, each node is initially assigned a current speed and direction. A node moves during a fixed interval of time with a constant speed and direction. After this time elapses, its speed and direction are updated. More specifically, the new speed and the new direction of a node at the nth time interval are computed as a function of the speed and direction of the same node at the $(n-1)$st interval, according to the following expressions: $v_n = \alpha v_{n-1} + (1-\alpha)\bar{v} + \sqrt{(1-\alpha^2)}v_{x_{n-1}}$; $\theta_n = \alpha\theta_{n-1} + (1-\alpha)\bar{\theta} + \sqrt{(1-\alpha^2)}\theta_{x_{n-1}}$, where v_n and θ_n are the new speed and direction of the mobile node at time interval n; $0 \leq \alpha \leq 1$ is the tuning parameter which defines motion randomness; $\bar{v}$ and $\bar{\theta}$ are constants that represent the desired mean speed and direction of the node; and $v_{x_{n-1}}$ and $\theta_{x_{n-1}}$ are random variables from a Gaussian distribution. In this model, totally random motion (Brownian motion) is obtained by setting the randomness parameter α to 0, and linear motion is obtained by setting α to 1.

We developed the **Modified Gauss-Markov** mobility model to describe node mobility in urban areas. In this model, we define two types of movement: (i) indoor with α_{in}, and (ii) outdoor with α_{out}, where typically $\alpha_{in} < \alpha_{out}$; this means that indoor nodes move more randomly than outdoor. Accordingly, we define indoor and outdoor areas. When a node hits a boundary between an indoor and outdoor area (or vice-versa), it changes the environment with probability β, and with probability $1 - \beta$ it bounces from the border and remains in the same environment. Whenever a node changes its environment, it resets its speed and direction of movement.

In the presented models, two nodes can establish a security association if they are in the *security range* of each other (for the fully self-organized network) or in each others' power range (in the authority based network). The security range is significantly smaller than the power range of mobile nodes and is the maximum range that is sufficient for the secure side channel to be set up.

5.3.2 Simulation parameters

Even if the models differ in the number of parameters and implementation, we try to introduce similar settings for all models to be able to compare the speed of establishment of the security associations between the models. In all simulations, we use the same simulation area, a 1000×1000 m square area (100×100 vertices in the case of grid) and we set the number of nodes to $n = 100$. We observe 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, in most cases, for the clarity of figures, we do not show the results obtained for the torus area.

In the Random Waypoint and the Restricted Random Waypoint simulations, the node maximum speed is set to 5 m/s or 20 m/s, and the minimum speed to 1 m/s [31]. The pause time is set to 100 s or 300 s.

For simulations with the Gauss-Markov mobility model, we set the randomness parameter α to $0.1,\ 0.5$ and 0.9, the speed standard deviation σ_v to 2 m/s and the angle standard deviation σ_θ to $\frac{\pi}{2 \cdot 1.96}$; This last value implies that 95% of the Gaussian random variables fall between $-\pi/2$ and $\pi/2$, because we want to avoid sharp turns even with a value of α close to 0. We choose the speed and the standard deviation in such a way that the model produces average speeds which are consistent with pedestrian movements, (i.e., between 0 and 5 m/s). Here, unlike in the original Gauss-Markov mobility model, the desired speed $\bar{v}$ and direction $\bar{\theta}$ are not fixed, but are recomputed in each step as an average value over all previous steps. These values are then reset every 200 steps.

The parameters for the Modified Gauss-Markov model were similar to the ones of the Gauss-Markov model, except that the randomness parameters differ for indoor and outdoor areas. Thus, $\alpha_{in} = 0.1$, $\alpha_{out} = 0.9$, $\sigma_{vin} = \sigma_{vout} = 2$ and $\sigma_{\theta in} = \sigma_{\theta out} = \frac{\pi}{2 \cdot 1.96}$. The transition probability between areas β is set to 0.25. The simulation area for this model is a 1000×1000 m square with bouncing boundaries and is divided into 100 grid areas squares of size 100×100 m each. In each of these grid areas, we center a local area of size 80×80 m or 20×20 m.

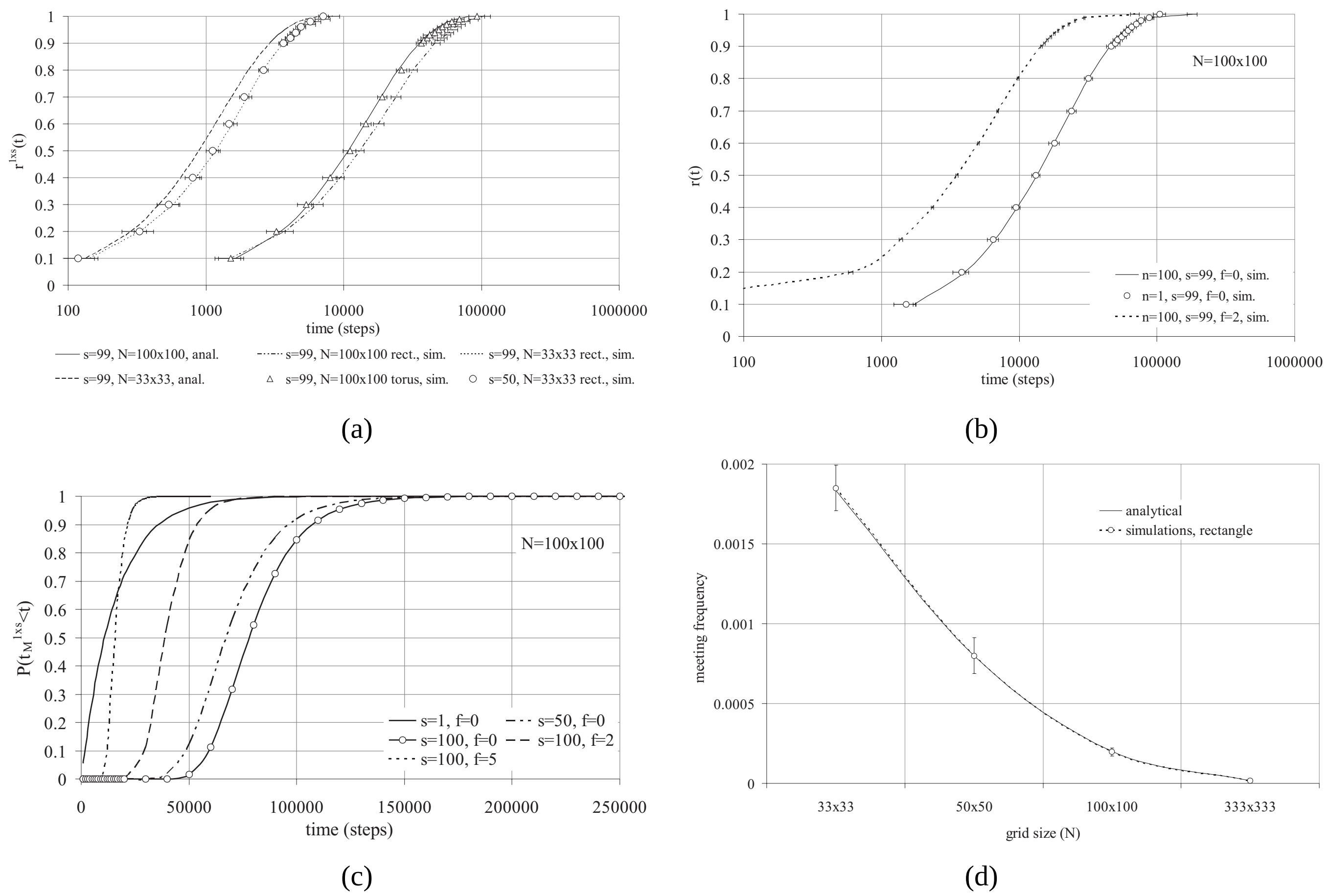
All simulations are run 20 times for each configuration and the averaged results are presented with 95% confidence intervals[9]. We note that even if all the models except random walk are continuous, we simulated them with a discrete time simulator written in C++.

5.4 Simulation results

Figure 2 shows the analysis and simulation results of the establishment of security associations with the random walk mobility model. Figure 2a plots the $r^{1 \times s}$ convergence for various simulation area sizes ($N = 33{\times}33, 100{\times}100$) and numbers of communication partners ($s = 50, 100$). The shape of the convergence curve shows that the majority of the security associations are established after a very short period of time, while it takes much longer for a node to establish security associations with all the communication partners of its interest. This figure also confirms our analytical results and shows that they can be used to approximate meeting times of nodes both in toroid and rectangular grid topologies. Figure 2b confirms our analytical findings that the convergence time and rate diminish proportionally with the number of friends. Figure 2c plots the convergence time distributions (cdf) of the $(1 \times s)$ meeting time for various values of s, computed analytically (Result 4). Figure 2d shows that our simulation results of the meeting frequency match our analytical observations (inter-meeting time, Result 7), that this frequency is inversely proportional to the size of the grid.

On Figure 3 we observe the convergence $r^{n \times s}(t)$ and the convergence time $t_M^{n \times s}$ with the Random Waypoint and the restricted waypoint mobility models, with and without the friends mechanism. Here again, like in the random walk model, we observe (Figure 3a) that the friends mechanism speeds up convergence proportionally to the number of friends. Furthermore, this figure demonstrates that, as expected, a higher average speed of nodes results in a faster convergence (and therefore a shorter convergence time). Figure 3b illustrates another very intuitive result: The convergence is faster if the nodes gather at and around meeting points. Figure 3c shows that, similarly to the random walk, in the Random Waypoint model, the convergence time depends mainly on the size of the area and not

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



(a)

(b)

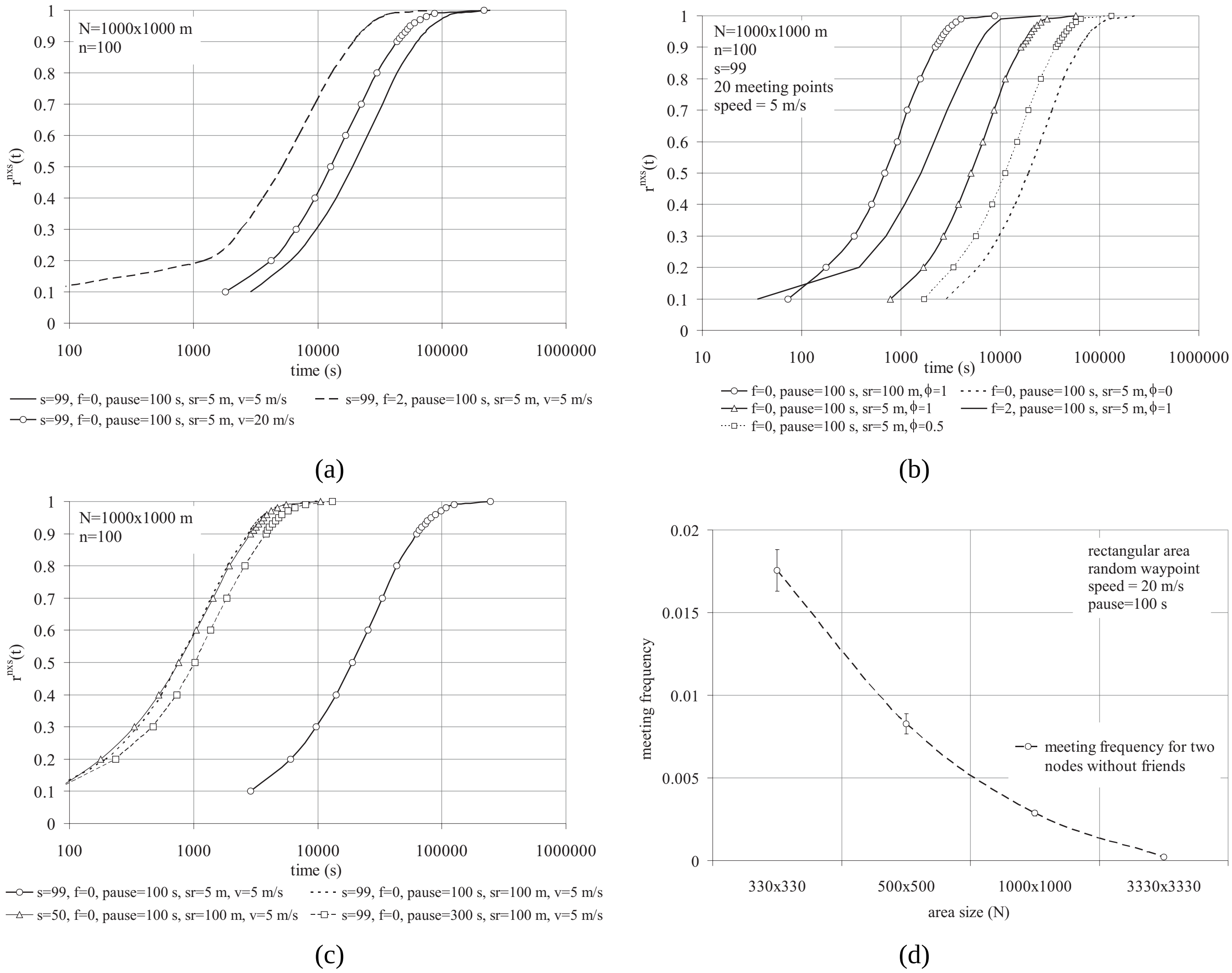
(c)

(d)

Figure 2. Random walk analysis and simulation results; a) The average $r^{1\times s}(t)$ convergence for various grid sizes and numbers of communication partners; b) Average $r^{1\times s}(t)$ and $r^{n\times s}(t)$ convergences for various grid sizes, with and without friends mechanism; c) The cumulative distribution function of $(1 \times s)$ convergence time, for various values of s; d) The meeting frequency of two nodes with Random Walk for different grid sizes.

on the number of the desired communication partners. This figure also illustrates that not only the speed but also the pause time can influence convergence, although not significantly. Furthermore, we observe that the automatic establishment of the security associations (authority based networks) is much faster than in the fully self-organized approach, because in the authority based networks, security associations can be established through the wireless channel, which is much larger that the secure side channel used to set up security associations in the self-organized case. Our further observations focus on meeting frequency between nodes. Figure 3d shows that with Random Waypoint mobility, the meeting frequency of two nodes is inversely proportional to the area size, like with the Random Walk mobility.

Finally, we observe the simulation results that we obtained with the Gauss-Markov and Modified Gauss-Markov models. From Figure 4a we observe that the more regular the movement of the nodes ($\alpha = 0.9$), the faster the establishment of security associations. We further observe the same effect of the friends mechanism on the convergence and convergence time as with the previously observed models. Figure 4b shows the convergence with the Modified Gauss-Markov mobility model. This figure indicates that in the presented "Manhattan" scenario, convergence is faster if nodes move around large indoor areas with narrow streets. As in the Random Waypoint scenario, with the Gauss-Markov and Modified Gauss-Markov mobility models convergence time remains between $10,000$ and $100,000$ seconds. However, if the nodes can establish security associations over wireless channels (authority based networks), the convergence is much faster and the convergence time is smaller than $10,000$ seconds.



(a)

(b)

(c)

(d)

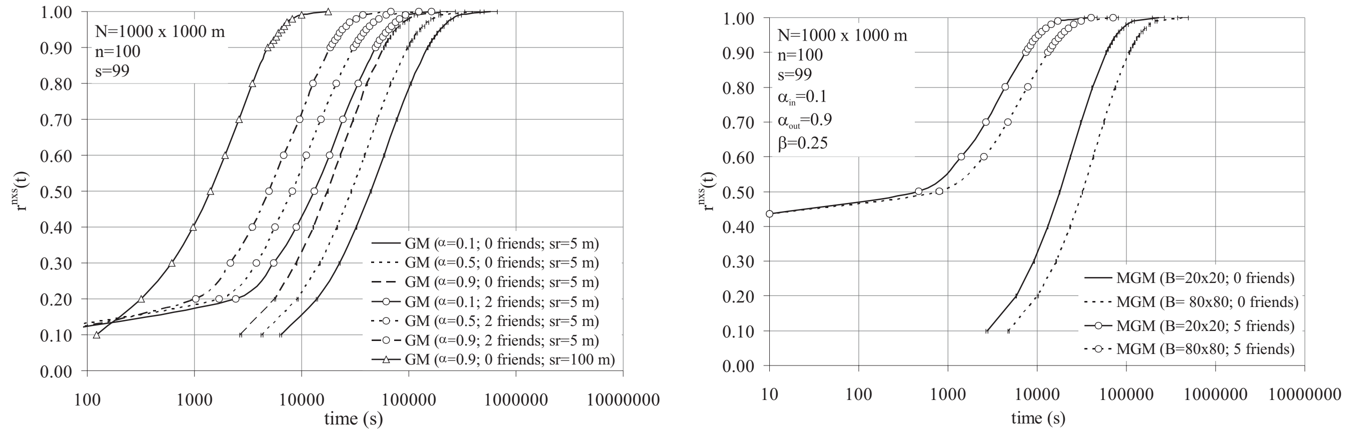
Figure 3. Random Waypoint and Restricted Random Waypoint simulation results; a) The average convergence with Random Waypoint with different speeds, numbers of friends and power (security) ranges; b) The average convergence with Restricted Random Waypoint with various restriction probabilities and power (security) ranges; c) The average convergence with Random Waypoint with different number of communication partners and pause times; d) The meeting frequency of two nodes with Random Waypoint for different area sizes.

6 CONCLUSION

In this paper, we have shown that mobility can help to provide security in mobile networks. We have illustrated our approach on two application scenarios in the area of mobile ad hoc networks: fully self-organized networks and networks with an off-line authority. In the first scenario, we have shown that the solution is intuitive to the users, as it mimics real life concepts (physical encounters and friends), and solves some classical problems of security in distributed systems. In the second scenario, a direct establishment of security associations over the (one-hop) radio link solves the well-known security-routing interdependency problem.

We have shown that our solution works both with public-key and with symmetric cryptography and we have provided the related protocols.

We have studied the pace of establishment of the security associations, both analytically and by simulations, under various mobility scenarios. In particular, we have extended the Random Waypoint model by introducing the concept of meeting points and we have enriched the Gauss-Markov model in order to include mobility in urban areas. We have observed that in the case of the random walk, where the node speed is constant, the convergence time is in the order of the $N \log N$, where N is the size of the observed area; but with the friend mechanism, this time can be significantly reduced (proportionally to the number of the friends). Similar convergence times can be



(a) (b)

Figure 4. a) Convergence with the Gauss-Markov mobility model; b) Convergence with the Modified Gauss-Markov mobility model.

observed both for the Restricted Random Waypoint and the modified Gauss-Markov models: for the considered scenarios (100 nodes, 1 km^2, 5 m/s, 0 friends, 100 m security range), the time necessary to establish security associations between all users is around $10,000$ seconds (less than 3 hours).

These durations 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 decide to move close to people of their interest. In ad hoc networks controlled by an (off-line) authority, nodes establish security associations automatically with other nodes in their power range, and the convergence time becomes significantly shorter; as our simulations showed, this time is ten to twenty times shorter than in the fully self-organized scenario. It is also worth noticing that in this case, more than 60% of the security associations are established within the first 1000 seconds.

As we mentioned in the Introduction, the approach proposed in this paper can in fact be used in *any* mobile network, 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 meet people of interest to them, as described in Section 3. A second example would be to secure email between PDAs communicating with each other via wireless LANs and the Internet. A third example would consist in securing spontaneous shared spaces. More generally, our solution can be perceived as an alternative (or, better, as a complement) to SPKI/SDSI, well suited to the case in which users are mobile.

To the best of our knowledge, this research effort is the first that shows how peer-to-peer security can be brought to mobile networks.

In the future, we plan to study even more realistic and more sophisticated mobility models, including those with correlated mobility patterns [2]. We will also study how an even *incomplete* set of security associations can be exploited to perform some crucial security operations. We also intend to further investigate rekeying and key revocation schemes. Finally, we intend to analyze the burden of the cryptographic functions on the processing units, especially in the public-key case.

7 Acknowledgments

We are thankful to Mario Čagalj, Mathias Grossglauser, Adrian Perrig and Milan Vojnović for useful comments and discussions. The authors also thank EPFL students David Schmalz and Olivier Roy for their contribution in producing simulation results. 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`).

REFERENCES

[1] N. Asokan and P. Ginzboorg. Key Agreement in Ad Hoc Networks. *Computer Communications*, 23:1627–1637, 2000.

[2] F. Bai, N. Sadagopan, and A. Helmy. IMPORTANT: A framework to systematically analyze the Impact of Mobility on Performance of RouTing protocols for Adhoc NeTworks. In *Proceedings of Infocom*, April 2003.

[3] L. Blažević, L. Buttyán, S. Čapkun, S. Giordano, J.-P. Hubaux, and J.-Y. Le Boudec. Self-Organization in Mobile Ad Hoc Networks: The Approach of Terminodes. *IEEE Communications Magazine*, June 2001.

[4] M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication, from Proceedings of the Royal Society, Volume 426, Number 1871, 1989. In *William Stallings, Practical Cryptography for Data Internetworks, IEEE Computer Society Press, 1996*. 1996.

[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*, 7(1), 2003.

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

[9] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age Matters: Efficient Route Discovery in Mobile Ad Hoc Networks Using Encounter Ages. In *Proceedings of MobiHoc*, 2003.

[10] R. Ellis. Torus Hitting Times Project, *(http://www.math.tamu.edu/∼rellis/comb/torus/torus.html)*.

[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] 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.

[14] B. Liang and Z. J. Haas. Predictive Distance-Based Mobility Management for PCS Networks. In *Proceedings of Infocom*, 1999.

[15] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proceedings of MobiCom*, 2000.

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

[17] G. Montenegro and C. Castelluccia. Statistically Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses. In *Proceedings of NDSS*, 2002.

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

[19] 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 MobiHoc*, 2003.

[20] D. B. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, December 1994.

[21] David J. Aldous and A. Fill. Markov chains on graphs. *manuscript under preparation*, 2000.

[22] M. Guerrero Zapata and N. Asokan. Securing Ad Hoc Routing Protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe)*, September 2002.

[23] 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.

[24] Peter G. Doyle and J. Laurie Snell. *Random walks and electric networks*. Carus Mathematical Monographs, No 22, 2001.

[25] Rahul C. Shah, Sumit Roy, Sushant Jain, and Waylon Brunette. Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks. In *Proceedings of the IEEE Workshop on Sensor Network Protocols and Applications (SNPA)*, to appear, 2003.

[26] Y.-C. Hu, A. Perrig, and D. B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *Proceedings of MobiCom*, September 2002.

[27] 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.

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

[29] 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.

[30] S. Čapkun, J.-P. Hubaux, and L. Buttyán. Mobility Helps Security in Ad Hoc Networks. In *Proceedings of MobiHoc*, 2003.

[31] J. Yoon, M. Liu, and B. Noble. Random Waypoint Considered Harmful. In *Proceedings of Infocom*, 2003.

[32] L. Zhou and Z. Haas. Securing Ad Hoc Networks. *IEEE Network*, 13(6):24–30, 1999.