Security Instantiation for Mobile Networks

Sandeep S. Kulkarni
Department of Computer Science and Engineering
Michigan State University
East Lansing MI 48824

Mohamed G. Gouda
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712

Anish Arora
Department of Computer and Information Science
Ohio State University
Columbus OH 43210

Abstract

In this paper, we focus our attention on the problem of assigning initial secrets to users in a mobile system so that they can use those secrets to ensure authentication and privacy during their communication. We present a protocol that maintains $O(\sqrt{n})$ secrets per user where n is the number of users in the system. We show that our secret distribution protocol suffices for privacy and authentication as well as secure multihop communication between two users. Furthermore, we note that the number of secrets maintained in this protocol is within a constant factor of the optimal solution for this problem. For the case where user capability prevents them from maintaining the necessary secrets, we propose a probabilistic protocol that maintains $O(log\ n)$ secrets per user. In this protocol, the probability of security compromise between two users is inversely proportional to the number of secrets they maintain.

Keywords : Mobile networks, Security, Instantiating security, Scalability

1 Introduction

Security is typically an important issue in mobile wireless networks where the communication medium is broadcast in nature and, hence, an adversary can overhear all messages sent by a user. For this reason, a sender must authenticate the receiver and encrypt any messages it sends. One way to achieve this authentication and encryption is to ensure that the sender and the receiver share a common secret that no other user in the network knows.

Another important issue in wireless networks is multihop communication. Specifically, if two users that need to communicate are not *close* to each other, they may require other users in the network to relay their messages. During such a relay, users may require that the intermediate users relaying the message cannot learn the contents of the message and that the intermediate users cannot generate messages that incorrectly appear to be from the sender.

In systems with a trusted server, the problem of distributing initial secrets is often handled by assigning each user a secret (e.g., password) that is known only to the user and the trusted server. In these systems (e.g., [1–4]), when two users interact, they use this trusted server to authenticate each other and to establish a secret that is known only to those two users. In many systems, especially in mobile systems, however, this approach is undesirable (respectively, impossible) as no trusted server is available *when two users need to communicate with each other*. Also, in these systems, using certificates signed by a trusted authority is undesirable, as the cost of encryption/decryption using them is often exorbitant. (As an illustration, on the MICA motes developed by UC Berkeley, encryption using public key is approximately 100-1000 times slower than the symmetric key encryption.) Thus, in these systems, the users must maintain sufficient secrets so that any two users that need to communicate can use these secrets for authentication and/or privacy.

Due to mobility, a user in a mobile network interacts with several users over a period of time [5]. Thus, over a period of time, a user should be able to interact with any user in the network. However, the mobile devices often have limited memory and limited computing ability. Therefore, the number of secrets they maintain and the amount of computation they perform should be small.

Based on the above discussion, it follows that one of the important problems for security in mobile networks is to establish a scalable approach for instantiating security so that each user maintains only a small number of secrets even though it can securely communicate (either with certainty or with high probability) with all users in the system.

[1]Email: sandeep@cse.msu.edu, gouda@cs.utexas.edu, anish@cis.ohio-state.edu

Web: `http://www.cse.msu.edu/~sandeep`, `http://www.cs.utexas.edu/~gouda`, `http://www.cis.ohio-state.edu/~anish`

Tel: +1-517-355-2387, +1-512-471-9532, +1-614-292-1836

This work was partially sponsored by NSF CAREER CCR-0092724, DARPA Grant OSURS01-C-1901, ONR Grant N00014-01-1-0744, NSF grant EIA-0130724, and a grant from Michigan State University.

We present two protocols for instantiating security in mobile systems. In the first protocol, the grid protocol, the user maintains $O(\sqrt{n})$ secrets where n is the number of users. This protocol ensures privacy and authentication between any two users that need to communicate with each other. We note that the number of secrets maintained in the grid protocol are within a constant factor of the optimal.

Since in large systems, users may not be able to maintain even $O(\sqrt{n})$ secrets, privacy and authentication cannot be guaranteed in them. For such systems, we present a protocol that provides probabilistic security, i.e., in this protocol, the level of security, the probability that the two parties can communicate using secrets that the intruder does not know, is proportional to the number of initial secrets that each user maintains.

Contributions of the paper. The main contributions of the paper are as follows:

- We present a protocol where each user maintains $O(\sqrt{n})$ secrets, where n is the number of users in the system. This protocol guarantees privacy and authentication.
- We also present a protocol for the case where the level of privacy is proportional to the number of initial secrets that each user maintains. In this protocol, the probability of a compromise is $a^{(O(m))}$, where $a < 1$ and m is the number of secrets that each user maintains. Thus, in this protocol, a small increase in the number of secrets maintained by a user substantially reduces the probability of privacy compromise. It follows that our probabilistic protocol is especially beneficial for the case where the users do not have the capability to maintain sufficient secrets to ensure privacy.

Organization of the paper. The rest of the paper is organized as follows: In Section 2, we precisely state the problem of instantiating security in mobile systems. Then, in Section 3, we present our protocol that provides privacy and authentication. In Section 4, we present our probabilistic protocol. In Section 5, we compare these protocols. Finally, we make concluding remarks in Section 6.

2 Model and Problem Statement

In this section, we identify the precise requirements for the initial secret distribution. Before we present these requirements, we consider different approaches that may be considered in static networks and argue that these solutions are not suitable for mobile networks.

One possible approach for obtaining privacy is to use certificates [6–8] and initially provide each user with a certificate signed by a trusted authority (respectively, a group of trusted authorities). Thus, when two users communicate, they can use these certificates to authenticate each other. However, as discussed in the Introduction, this solution requires high computing power. The drawback of this solution suggests that we should use shared secrets instead of certificates. We consider two simple protocols that use such shared secrets: *single secret protocol* and *full secret protocol*.

Single secret protocol. The simplest protocol that ensures that the sender and the receiver share at least one secret is to establish a single shared secret that all *legitimate* users in the system know. It follows that intruders outside the system cannot learn about the communication between legitimate users. While this approach is currently used in existing sensor networks [9], it is clear that in this protocol, the compromise of one user compromises the security for all.

Full secret protocol. Another solution for sharing secrets between a sender and a receiver is to establish a separate shared secret between every pair of users. Although, in this protocol, the compromise of one user does not affect others, each user needs to maintain $n-1$ secrets where n is the number of users in the system. Hence, this solution becomes infeasible when the number of users is large and memory associated with each user is low.

Clearly, if we require that the secret shared between two users, say j and k, is not known to any other user in the system, then each user must maintain $n-1$ secrets where n is the number of users. To reduce the number of secrets, we allow j and k to share a collection of secrets, and require that no other user in the system knows all the secrets in that collection. In this situation, it would be possible for j and k to use a combination of these secrets (e.g., by xor-ing these secrets, or by applying some one way hash function to the collection of shared secrets) to ensure privacy and authentication. Thus, we define the problem statement for guaranteed security as follows:

Problem for *guaranteed* security. Design a secret distribution protocol such that given any two users j and k, they share a collection of secrets such that no other user in the system knows all the secrets in that collection. □

Likewise, we define the problem of probabilistic security as follows:

Problem for *probabilistic* security. Design a secret distribution protocol such that given three randomly selected users, j, k and l, the probability that there is a secret (respectively, a collection of secrets) that both j and k know but l does not know is proportional to the number of initial secrets that j and k have. □

Intruder Model. We assume that an intruder is a single non-colluding device that does not share its secrets with any other user. We address the issue of collisions in the Conclusion.

3 Guaranteed Security Protocol

In this section, we present the first protocol, the *grid protocol* that solves the problem of guaranteed security. We first describe the secret distribution protocol that identifies how the secrets are distributed and then identify the protocol that allows nodes to choose a secret they can use in private communication.

Secret distribution protocol. In this protocol, each user has two types of secrets: *direct secrets* and *grid secrets*: A *direct secret* is shared between exactly two users and each *grid secret* is shared among multiple users. As the name suggests, the *grid secrets* are arranged in a 2-D grid. For example, see Figure 1 where 16 grid secrets are arranged in a 4x4 grid. Each node in the grid is also associated with a user. Each user gets the *grid secrets* associated with the nodes in its row and in its column. Thus, in Figure 1, user $u_{\langle 2,3\rangle}$ gets the secrets $k_{\langle 1,3\rangle}$, $k_{\langle 2,3\rangle}$, $k_{\langle 3,3\rangle}$, $k_{\langle 4,3\rangle}$, $k_{\langle 2,1\rangle}$, $k_{\langle 2,2\rangle}$, and $k_{\langle 2,4\rangle}$.

Each user maintains *direct secrets* with users in its row and in its column. Thus, in Figure 1, $u_{\langle 2,3\rangle}$ maintains a separate *direct secret* with $u_{\langle 1,3\rangle}$, $u_{\langle 2,3\rangle}$, $u_{\langle 3,3\rangle}$, $u_{\langle 4,3\rangle}$, $u_{\langle 2,1\rangle}$, $u_{\langle 2,2\rangle}$, and $u_{\langle 2,4\rangle}$. (The *direct secrets* are not shown in the figure.)

Secret selection protocol. When two users $u_{\langle j_1,k_1\rangle}$ and $u_{\langle j_2,k_2\rangle}$ need to communicate, they use the following protocol:

If $(j_1 \neq j_2 \ \wedge \ k_1 \neq k_2)$ // Different row and column
  Use the *grid secrets* $k_{\langle j_1,k_2\rangle}$ and $k_{\langle j_2,k_1\rangle}$
Else // Same row or column
  Use the *direct secret* between $\langle j_1, k_1\rangle$ and $\langle j_2, k_2\rangle$

Theorem 3.1 The grid protocol solves the problem of guaranteed security.

Proof. We observe that when $\langle j_1, k_1\rangle$ and $\langle j_2, k_2\rangle$ communicate, the collection of secrets they use is not known to any other user in the system. Thus, using those secrets, these two users can achieve privacy and/or authentication. □

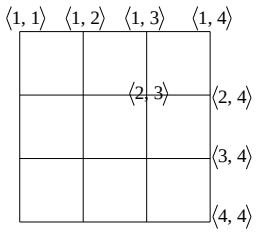

Figure 1: Grid protocol: A node marked $\langle j, k\rangle$ is associated with user $u_{\langle j,k\rangle}$ and *grid secret* $k_{\langle j,k\rangle}$

Remark. In the grid protocol, the user identity is specified by its location in the grid. This location need not be secret and, hence, when two users, say Alice and Bob, want to talk they can simply reveal their own location in the grid. In other words, a user need not maintain any information about the grid except its own location.

Theorem 3.2 In the grid protocol, each user maintains $2\sqrt{n} - 1$ *grid secrets* and $2(\sqrt{n}-1)$ *direct secrets*. □

We note that the number of secrets maintained per user in the grid protocol are within a constant factor of the minimum number of secrets necessary to ensure guaranteed security. We show this result by using the fact that a user needs to share a collection of secrets with every other user in the system and that this collection should not be available to any other user in the system. Due to space limitations, we relegate this result to [10].

4 Probabilistic Security Protocol

Based on the lower bound identified in Section 3, if users in a mobile system cannot maintain $O(\sqrt{n})$ secrets, then privacy and authentication cannot be guaranteed. To deal with this negative result, we propose solutions for probabilistic security where the probability of privacy compromise is inversely proportional to the number of secrets that users maintain. Before we discuss this protocol, we first introduce the notion of *effectiveness* for a secret distribution protocol.

Definition (Effectiveness). We say that a secret distribution protocol is $\langle s, p\rangle$ effective if the maximum number of secrets that any user has is s and given three randomly selected users, j, k and l, the expected probability that l knows the secret (respectively, all the secrets) used by j and k is at most p.

Definition (Storage dominate). Given two secret distribution protocols, pr_1 and pr_2, with effectiveness $\langle s_{pr_1}, p_{pr_1}\rangle$ and $\langle s_{pr_2}, p_{pr_2}\rangle$ respectively, we say that pr_1 *storage dominates* pr_2 if $s_{pr_1} \leq s_{pr_2}$.

Definition (Security dominate). Given two secret distribution protocols, pr_1 and pr_2, with effectiveness $\langle s_{pr_1}, p_{pr_1}\rangle$ and $\langle s_{pr_2}, p_{pr_2}\rangle$ respectively, we say that pr_1 *security dominates* pr_2 if $p_{pr_1} \leq p_{pr_2}$.

Definition (Dominate). Given two secret distribution protocols, pr_1 and pr_2, with effectiveness $\langle s_{pr_1}, p_{pr_1} \rangle$ and $\langle s_{pr_2}, p_{pr_2} \rangle$ respectively, we say that pr_1 *dominates* pr_2 if $s_{pr_1} \leq s_{pr_2}$ and $p_{pr_1} \leq p_{pr_2}$.

Observation 4.1

- The single secret protocol is $\langle 1, 1 \rangle$ effective.
- The full secret protocol is $\langle n-1, 0 \rangle$ effective.
- The grid protocol is $\langle 4\sqrt{n} - 3, 0 \rangle$ effective
- The single secret protocol storage dominates the full secret protocol.
- The full secret protocol security dominates the single secret protocol.
- The grid protocol dominates the full secret protocol. □

Thus, the single secret protocol and the full secret protocol (respectively, grid protocol) are two extremes for security distribution protocols. In this section, we focus on identifying a protocol that is $\langle x, y \rangle$ effective where $1 < x < O(\sqrt{n})$ and $0 \leq y < 1$. Moreover, we focus on a security protocol where the level of security is adaptive, i.e., if the number of secrets that each user gets is increased then the probability of a compromise is reduced. We present one such protocol, next. In this protocol, we focus only on the issue of privacy. For reasons of space, the issue of authentication is relegated to [10].

First, in Section 4.1, we present the version of the tree protocol where only one tree is used. Then, in Section 4.2, we present the version where multiple trees are used.

For each of these versions, we first identify the secret distribution protocol that determines the secrets that each user should get. Then, we present the secret selection protocol; when two users need to communicate, they use this protocol to determine a shared secret that they should use. Subsequently, we compute the probability of compromise.

4.1 Single Tree Protocol

Secret distribution protocol. We organize the secrets in a tree (cf. Figure 2). Each non-leaf node is associated with a secret and each leaf is associated with a user. Each user is provided the secrets along the path towards the root. Thus, user u_1 has the secrets, k_1, k_2 and k_4.

Secret selection protocol. When two users, say, j and k, want to communicate, they use the secret associated with their ancestor. For example, if users u_1 and u_2 want to communicate then they will use k_4 whereas if users u_1 and u_5 want to communicate then they will use k_1.

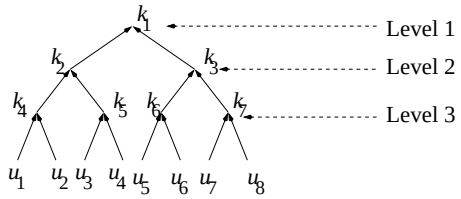

Figure 2: Tree Protocol

Computing the probability of security compromise. Let l be an intruder that can observe the communication between j and k. We identify the probability that l is aware of the secret that j and k use. During this analysis, let the degree of the secret-tree be d.

Now, we consider different cases based on the shared secret that j and k use during communication. First, we consider the probability that j and k use the secret at the root (level 1). The probability of this case is $\frac{d-1}{d}$. And, in this case, the probability that l is aware of the secret that j and k use is 1; all users in the secret-tree have the secret associated with the root.

Next, we consider the probability that j and k use the secret at level 2 in the tree. Such a situation arises if k is a descendant of the level-2-ancestor of j *and* k is not a descendant of the level-3-ancestor of j. Thus, the probability of this case is $\frac{1}{d} * \frac{d-1}{d}$. Moreover, l is aware of the shared secret between j and k iff l is a descendant of the level-2-ancestor of j. Thus, the probability of this case is $\frac{1}{d}$. Continuing thus, the probability, $p_{compromise}$, that l is aware of the shared secret used by j and k is:

$$\begin{aligned} p_{compromise} &= \frac{d-1}{d}.1(\Sigma_{j=0}^{h}(\tfrac{1}{d})^{2j}) \\ &< \frac{d-1}{d}.1(\Sigma_{j=0}^{\infty}(\tfrac{1}{d})^{2j}) \\ &= \frac{d-1}{d}\frac{1}{1-\frac{1}{d^2}} \\ &= \frac{d}{d+1} \end{aligned}$$

Theorem 4.2 The single tree protocol is $\langle log_d(n), \frac{d}{d+1} \rangle$ effective. □

4.2 Multiple Tree Protocol

In the single tree protocol, the probability of security compromise is minimized when $d=2$. With $d=2$, when j and k want to communicate with each other, there is a $\frac{2}{3}$ probability that a third user, l, knows the secret used by j and k. We can reduce this probability further by using multiple secret-trees. We discuss the secret distribution and secret selection protocol with such multiple trees, next.

Secret distribution protocol. In this protocol, the secrets are arranged in multiple trees. Similar to the

single tree protocol, in this protocol, each internal node in each tree is associated with a secret and each leaf is associated with a user. Each tree includes all users. For each tree, the user gets the secrets associated with its ancestors in that tree.

Secret selection protocol. For each secret-tree, j and k identify the secret associated with their least common ancestors. Then, they use the combination of all these secrets (e.g., by xor-ing these secrets or by passing those secrets through a one way hash function [11]) during communication. It follows that l can learn the communication between j and k iff l knows all these secrets.

Clearly, if we use two trees where the position of all users is identical and if l knows the secret (used by j and k) in the first tree then, by definition, l will know the secret in the second tree. Hence, when we use multiple trees to reduce the probability of compromise the probability that l knows the secret in one secret-tree should be independent of the probability that l knows the secret in another tree. This can be achieved if there is no correlation between the location of a user across two trees.

Given K of secret-trees, each with degree d, the probability that l knows secrets from all the trees is $(\frac{d}{d+1})^K$. Thus, we have

Theorem 4.3 The multiple tree protocol with K trees is $\langle Klog_d(n), (\frac{d}{d+1})^K)\rangle$ effective. □

Remark. Note that the above result is applicable for the case where $K << n$. If the number of trees is close to n then the lack of correlation is not possible. Moreover, if the number of trees is close to the number of users then the users would need to maintain a large number of secrets.

As an example, consider the case where we have 2^{20} users and 20 secret-trees of degree 2 are maintained. In this case, the probability of a compromise is $3 * 10^{-3}$ when users maintain only 400 secrets. By contrast, a user would need over a million secrets if we were to maintain a separate secret between every pair of users.

Extension: Users with different capabilities. To extend our algorithm for the case where users have different capabilities, we proceed as follows: Based on the maximum capability of any user, we identify the number of trees used. A user with lower capability will only maintain keys from the first few trees. Now, if j and k communicate and j has lower capabilities than k then they will use only secrets from those trees for which j has maintained the secrets.

5 Protocol Comparison

In this section, we compare the protocols presented in Sections 3 and 4. These results show that a reasonable level of security can be obtained by maintaining a small percentage of secrets maintained by the the grid protocol. (For reasons of space, additional performance results are relegated to [10].)

Since the number of secrets in the grid protocol are within a constant factor of the minimum number of secrets that users need to maintain, it follows that the probabilistic protocol is especially useful to provide a reasonable level of security by maintaining a small number of secrets.

Analytical modeling of probabilistic protocol. Suppose that a user maintains m secrets in the tree protocol. Thus, $\frac{m}{log_2 n}$ trees are maintained (For simplicity, we ignore the issue of partial trees). The level of security with m secrets is $(\frac{2}{3})^{(\frac{m}{log_2 n})}$

Tradeoff between guaranteed security and probabilistic security. Figure 3 compares the cost of deterministic security versus probabilistic security. More specifically, we ask the question: *How much security could be obtained by using the tree if we maintain only a certain percentage of secrets maintained by the grid protocol?* Figure 3 compares the tree protocol and the grid protocol. Based on this graph, we observe that for a group of 10,000 users, maintaining only 20% of the secrets is sufficient to ensure that the probability of security compromise is approximately 6%. Further, when the number of users is large, the percentage of secrets required for the same level of security is further reduced.

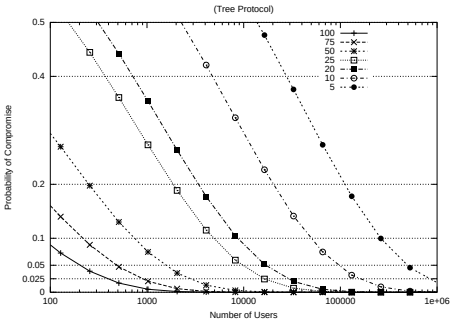

Figure 3: Probability of compromise if we maintain a certain percentage of secrets used by the grid protocol

6 Conclusion

In this paper, we presented two protocols for instantiating security in mobile networks where each user begins with a set of *initial secrets*. These protocols allow users to obtain privacy and authentication while communicating with each other. Moreover, when two users communicate over a multi-hop path, they can ensure that intermediate users can neither learn the contents of the transmitted messages nor generate messages that incorrectly appear to originate from the sender.

First, we presented the *grid protocol* that solved the problem of guaranteed security, i.e., it ensured that when two users, say j and k, communicate, the set of secrets they use is not known to any other user. This protocol maintained $O(\sqrt{n})$ secrets where n is the number of users in the system. Also, the number of secrets maintained by the grid protocol is within a constant factor of optimal.

Based on the optimality of the number of secrets maintained by the grid protocol, when the number of users is large and users cannot maintain all these secrets, their only choice is to provide probabilistic authentication and privacy. For these cases we presented a probabilistic protocol where the probability of a security compromise is proportional to the secrets they can maintain. In this protocol, the number of secrets maintained by a user is $O(log\ n)$, where n is the number of users. Moreover, as shown in Section 5, in the probabilistic protocol, maintaining a small number of secrets ensured that the probability of security compromise is low.

Although we assumed that multiple users do not combine their secrets compromise security, our protocols do indeed offer protection from such collusion. Specifically, in the grid algorithm, if two users collude then they can compromise communication between at most $O(\sqrt{n})$ users. Moreover, by maintaining multiple grids (as done in probabilistic protocols), this probability can be reduced further. Note that as long as the number of such grids is small, the number of secrets maintained by a user is $O(\sqrt{n})$, which is much less compared to the full secret protocol. Likewise, the probabilistic protocol can also resist security violations in the presence of colluding users.

The protocols presented in this paper are also applicable in sensor networks where a large number of sensors are deployed to achieve a common task (e.g., monitoring). In these networks, each sensor communicates with a small subset of its neighbors even though the number of sensors is often very large. However, at the time of manufacturing (respectively, programming) of the sensors, it is difficult to determine the neighbors of a sensor after deployment (e.g., by dropping them from a plane). Thus, our probabilistic protocol for managing the probability of security compromise based on the capabilities of sensors will be especially useful in this context.

For reasons of space, we did not discuss *how* the user obtains the initial secrets as this issue is orthogonal to the issue of *what* secrets a user should get. A user may obtain these initial secrets in several ways, e.g., a user may obtain these secrets by initially visiting (respectively, periodically revisiting) a trusted server. Towards this end, each user can maintain a secret that is only known to that user and the server. In such a situation, the server would need to maintain several secrets. However, this is likely to be acceptable as this trusted server will typically have more capability than the user itself. Of course, this secret cannot be used when two users communicate as they may not be able to reach the server. Alternatively, in the context of sensor networks, these secrets could be statically added to a sensor when it is programmed/deployed.

References

[1] A. Perrig, R. Szewczyk, J. Tyger, V. Wen, and D. Culler. Spins: Security protocols for sensor networks. *Wireless Networks*, 8:521–534, 2002.

[2] M. Tatebayashi, N. Matsuzaki, and D.B. Newman Jr. Key distribution protocol for digital mobile communications systems. *Advances in Cryptology*, 1990.

[3] V. Varadharajan and Y. Mu. Design of secure end-to-end protocols for mobile systems. *Wireless*, 1996.

[4] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of ACM*, 21:993–999, 1978.

[5] M. Grossglauser and D. Tse. Mobility increases the capacity of adhoc wireless networks. *IEEE/ACM Transactions on Networking*, 10(4):477–486, 2002.

[6] J. Kong, P. Zefros, H. Luo, S. Lu, and L. Zhang. Providing robust and ubiqutious security support for mobile ad-hoc networks. *IEEE International Conference on Network Protocols*, 2001.

[7] J. Hubaux, L. Buttyan, and S. Capkun. The quest for security in mobile ad-hoc networks. *ACM Symposium on Mobile Ad Hoc Networking & Computing*, 2001.

[8] L. Zhou and Z. Haas. Securing ad hoc networks. *IEEE Network*, 13(6), 1999.

[9] Chris Karlof, Naveen Sastry, and David Wagner. Tinysec: Link layer security for tiny devices. Available at `http://www.cs.berkeley.edu/~nks/tinysec/`, 2003.

[10] S. S. Kulkarni, M. Gouda, and A. Arora. Security instantiation for mobile networks. Technical report, Michigan State University, 2003.

[11] R. Rivest. The MD5 message-digest algorithm. *RFC 1321, Internet Activities Board*, 1992.