Monitoring-Based Key Revocation Schemes for Mobile Ad Hoc Networks: Design and Security Analysis

Katrin Hoeper*,[a], Guang Gong[b]

[a] *Computer Security Division, National Institute of Standards and Technology (NIST), Gaithersburg, MD 20878, USA*
[b] *Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada*

Abstract

This article proposes a parameterized trust model and security analysis framework for monitoring-based schemes that enables the identification of malicious nodes in mobile ad hoc networks (MANETs) and other decentralized networks. We utilize these results to design a practical decentralized key revocation scheme for MANETs in which nodes monitor their neighbors and securely propagate their observations, where security thresholds δ and ε protect against false accusations from groups of malicious nodes. Our revocation scheme is the first of its kind to take the inaccuracy of the underlying monitoring scheme into account. For example, we derive upper bounds for false positive rate α and false negative rate β that ensure the functionality and security of the revocation scheme. By utilizing security parameters δ, ε, α and β, we provide an extensive security analysis showing how to select these and other system parameters to mitigate a wide range of attacks from insiders and outsiders as well as colluding nodes.

Key words: Mobile ad hoc networks, security, monitoring, key revocation, identity-based cryptography.

*Parts of this work were conducted while the author was with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada.

Email addresses: khoeper@nist.gov (Katrin Hoeper), ggong@calliope.uwaterloo.ca (Guang Gong)

1. Introduction

Due to their versatility, decentralized infrastructure and other unique properties, there has been a growing interest in mobile ad hoc networks (MANETs) for military, government and commercial applications. A primary security challenge in such networks is the likelihood of node compromises caused by weak physical protection and potentially hostile environments. This means that key revocation mechanisms are essential and nodes must be capable of instantly verifying whether the key of another node has been revoked. Because of the decentralized nature of MANETs, revocation schemes must be based on the nodes' own observations. Initial ideas for MANET revocation schemes have been outlined elsewhere [14, 21] and, more recently, a few practical monitoring-based revocation schemes have been proposed [5, 6, 9, 11, 16, 20].

In this article, we introduce a novel parameterized trust model and security analysis framework for monitoring-based key revocation schemes in MANETs. This work is the first to study the impact of the inherent inaccuracy of monitoring schemes on the functionality and security of the revocation scheme itself. Clearly, without considering false positive and false negative rates, α and β, the analysis of a monitoring-based revocation scheme's resilience to collusive attacks (e.g. by unidentified malicious nodes) becomes meaningless. For example, we derive upper bounds for β that mitigate collusive attacks. In addition, we demonstrate that only a single threshold δ may not be sufficient to thwart Sybil attacks, and hence, introduce a second threshold ε as a solution. Additional parameters are propagation range m and expiry interval ΔT that affect the scheme's performance. These parameters can be selected to reduce the impact of roaming adversaries that are trying to escape their key revocations. In addition to Sybil and roaming attacks, our threat model addresses modification, fabrication, impersonation, battery exhaustion and replay attacks, as well as several advanced collusive attacks. The proposed trust model and threat model as well as the derived security parameters, parameter relations and boundaries can be easily applied to any accusation-based revocation scheme and in fact to any monitoring-based scheme in decentralized networks.

We apply our trust and threat models—with all the above-mentioned security parameters—to the accusation-based revocation scheme in [10]. We chose the scheme in [10] because it follows the recent trend of utilizing the efficient key management of pairing-based identity-based cryptographic (IBC)

schemes to secure MANETs [7, 10, 11, 13, 20]. Among the proposed IBC-based schemes, [10] uniquely provides detailed algorithm descriptions, utilizes two security thresholds, cryptographically protects messages without the need of computationally demanding digital signatures, enables efficient and secure reception of previous accusations, as well as the revocation of the nodes' own keys. Applying the trust and threat model to the original scheme helped us to identify several security flaws (not detailed here) and generate a modified scheme that addresses all flaws. For example, the original scheme required nodes to identify neighbors in m-hop range, which may be impractical in some applications. This requirement no longer exists in the revocation scheme presented here. Furthermore, vulnerabilities to replay, battery exhaustion and roaming attacks, present in the original scheme, are addressed and numerous extensions to the scheme are presented. As a result, we describe a practical, completely decentralized accusation-based revocation scheme for MANETs, together with its security analysis which provide several parameters to fine-tune the scheme's security and performance, depending on the application and network environment.

The remainder of this article is organized as follows. In the next section, we briefly review background information and related work. In Section 3, we discuss the system set-up and trust model. The key revocation scheme is presented in Section 4 and analyzed in Sections 5 and 6. Finally, we derive conclusions in Section 7.

2. Background and Related Work

In this section, we briefly review pairing-based IBC schemes, as well as existing revocation and monitoring schemes for MANETs.

2.1. Pairing-based IBC Schemes

The first ID-based encryption scheme from the Weil pairing was introduced by Boneh and Franklin [2]. In the following, we give a definition of cryptographic bilinear mappings, the building block of any pairing-based IBC scheme.

Let $\mathbb{G}_1$, $\mathbb{G}_2$ be two groups of the same prime order q. $\mathbb{G}_1$ is as an additive group, whereas $\mathbb{G}_2$ is a multiplicative subgroup of a finite field. Let P be an arbitrary generator of $\mathbb{G}_1$. Assume that the discrete logarithm problem (DLP) is hard in $\mathbb{G}_1$ and $\mathbb{G}_2$. A bilinear mapping $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$ must satisfy the following properties:

- *Bilinearity:* $\hat{e}(aP, bQ) = \hat{e}(P, Q)ab$ for all P, $Q \in \mathbb{G}_1$ and a, $b \in \mathbb{Z}_q^*$.
- *Non-degeneracy:* If P is a generator of $\mathbb{G}_1$, then $\hat{e}(P, P)$ is a generator of $\mathbb{G}_2$. In other words, $\hat{e}(P, P) \neq 1$.
- *Computable:* There exists an efficient algorithm to compute $\hat{e}(P, Q)$ for all P, $Q \in \mathbb{G}_1$.

The security of pairing-based IBC schemes is based on the so-called Bilinear Diffie-Hellman Problem (BDHP) [2] which is defined as follows.

Definition 1 (Bilinear Diffie-Hellman Problem (BDHP)). *Given two groups $\mathbb{G}_1$ and $\mathbb{G}_2$ of the same prime order q, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \mapsto \mathbb{G}_2$ and a generator P of $\mathbb{G}_1$, the BDHP is to compute $\hat{e}(P, P)^{abc} \in \mathbb{G}_2$ for any $a, b, c \in \mathbb{Z}_q^*$ given (P, aP, bP, cP).*

In any IBC scheme, the key generation center (KGC) is responsible to select the system parameters and set up the system such that the BDH problem is hard.

2.2. Revocation in MANETs

Despite its importance, many PKI and IBC-based security solutions for MANETs ignore key revocation or just briefly outline an idea for a solution without providing actual algorithms, e.g. [7, 12, 13, 14, 21]. To date only a few practical and comprehensive revocation schemes for MANETs have been proposed [5, 6, 9, 11, 16, 20].

The revocation scheme in [6] employs an accusation scheme with threshold δ. Accusations are frequently broadcasted throughout the network. Newly joining nodes receive the unprotected accusation tables from all other nodes and accept accusations from senders with sufficiently large trust value, where trust values depend on the number of accusations a node has made and/or received and other parameters.

In [5], a revocation scheme for sensor networks with pre-distributed secret keys is presented. Revocations are based on monitoring neighbors but rather than counting accusation towards a threshold, accusors broadcast secret shares. If more than δ secret shares have been broadcasted, nodes can reconstruct the secret and thus know that the owner's key is revoked.

Instead of revoking keys, the authors of [16] propose that nodes need to be reelected by a group of nodes to obtain a new key, where reelection is based

on observed behavior. Like [5], the reelection mechanism is based on secret key sharing. To avoid costly secret key reconstructions, the authors in [16] also outline a lightweight scheme in which nodes periodically broadcasts a so-called buddy-list with identifiers of trusted nodes. A third, very radical approach in [16] is to revoke the keys of accusor and accused node.

In [20], a distributed on-line KGC consisting of n network nodes (called D-KGC) carries out key revocations and renewals. The distributed KGC is implemented using a (k, n)-threshold scheme. Nodes monitor their neighborhood and send their accusations to b assigned D-KGCs. Once a threshold δ is reached, a group of k D-KGCs collaboratively signs a revocation message.

In [9], nodes predict the behavior of other nodes based on their own observations and reports from neighbors utilizing a 3-dimensional Dirichlet distribution. Here nodes can have three different states: trustworthy, suspicious and malicious, which enables nodes to make multilevel responses.

In [11], nodes monitor their neighbors and send their accusations to an m-hop neighborhood. Accusations are counted towards a δ-threshold, whereas reported accusations are additionally protected by an ε-threshold. Several features as well as limitations of the scheme have been outlined in Section 1, and the limitations will be addressed in our revocation scheme in Section 4.

Concluding, none of the discussed revocation scheme considers the false positive and false negative rates, α and β, of the monitoring scheme which is crucial for the security analysis of numerous attacks especially by colluding adversaries.

2.3. Misbehavior Detection Schemes

The following metrics that have been proposed for detecting malicious nodes in MANETs are suitable for monitoring-based revocation schemes:

- count number of dropped packets [3, 4, 15]
- count number of generated packets [1]
- use anomaly detection systems to detect unusual behavior [19]
- use intrusion detection systems to detect known attacks [15]

Whenever the threshold of the applied metric is reached, the node is marked as malicious and an accusation message is sent. Similarly, the monitoring results can be used to identify honest nodes as necessary in rewarding schemes [16].

3. System Set Up and Trust Model

In this section, we briefly describe the system set up, assumptions, notations and key revocation list (KRL) creation of the presented revocation scheme. Finally, we define the trust model.

3.1. Set Up IBC Scheme

For the proposed revocation scheme, we assume the existence of a pairing-based IBC framework. Here, we assume an external KGC for key generation and distribution. The system set up is the same as described in [11]. In particular, the IBC scheme has $params = \langle q, \mathbb{G}_1, \mathbb{G}_2, \hat{e}, P, P_{pub}, H_1, H_2 \rangle$ as public parameters, whereas the KGC's long-term private key s, also referred to as master key, is kept confidential. Each node i in the network has a public key

$$Q_i(t_x, v_i) = H_1(ID_i||t_x||v_i), \tag{1}$$

where ID_i is the identifier of node i, t_x the expiry date of the key and v_i the version number of the key. Version numbers start with $v = 1$ for every new expiry date t_x and are incremented with each key renewal that occurs before t_x. This key format allows instant key renewal [11]. The KGC computes the private key $d_i = sQ_i$ for each node i in the network using the system's master key s and distributes it to i. Whenever two network nodes i and j wish to communicate for the first time, they each compute their pairwise pre-shared key K_{ij} according to

$$K_{ij} \quad = \quad \hat{e}(d_i, Q_j) = \hat{e}(d_j, Q_i), \tag{2}$$

and store the key for future correspondences.

3.2. System Assumptions

The system assumptions for our key revocation scheme can be summarized as follows:

1. bidirectional communication links
2. each node has a monitoring scheme implemented
3. each node i has a unique identity ID_i
4. each node knows identities and hop-distance of its one-hop neighbors

Table 1: List of Notations for Key Revocation Scheme

N, $\Omega = \|N\|$	Set and number of all network nodes
N_i, Ω_i	Node i's perception of set and number of network nodes
$N_{1,i}$, $\sigma_i = \|N_{1,i}\|$	Set and number of i's one-hop neighbors
$\overline{\sigma}$, $\overline{\Omega}$	Average number of one-hop neighbors and network nodes
δ, ε	Thresholds for revocation and reported accusations
m	Propagation range of accusation messages
$\mathcal{KRL}^i(t_x)$	i's key revocation list for keys with expiry date t_x
t_x, ΔT	Expiry date and expiry interval, i.e. $t_{x+1} = t_x + \Delta T$
$\|x - y\|$	Euclidian distance between two nodes x and y
α, β	False positive & false negative rates of monitoring scheme
$a\|\|b$	Concatenation of two binary strings a and b
$\underline{c}^i_j$	Column vector in $\mathcal{KRL}^i$ with accusations from node j
$\underline{r}^i_j$	Row vector in $\mathcal{KRL}^i$ with accusations against node j
r_k	Counter for received k-vectors

5. each node i obtains a private and public key pair (d_i, Q_i) from an external KGC

The first two assumptions enable nodes to monitor their neighbors in communication range. We discussed some metrics for monitoring schemes in Section 2.3 and assume that a suitable scheme is implemented on each node. Assumptions 3 and 4 are necessary to unambiguously identify network nodes and one-hop neighbors, respectively. To identify all one-hop neighbors, a node could scan all nodes in communication range and/or send out hello messages and wait for replies. Assumption 5 is necessary because cryptographic keys are used for message protection in our revocation scheme. Here we assume an external off-line KGC that only distributes keys to nodes that have successfully authenticated to the KGC and are authorized to join the network.

3.3. Notations

We need to introduce some notations for our scheme. A summary of notations and symbols can be found in Table 1. Let N denote the set of all

network nodes, where $\Omega = |N|$ is the number of network nodes. We use Ω_i to denote the number of nodes node i currently possesses information about (such as identity, public key and/or accusation messages from or against this node) and use N_i to describe the set of these nodes. Note that Ω, N, $|Omega_i$ and N_i are all variable due to the dynamic character of MANETs. R is the communication range for transmitting and receiving messages and assumed to be constant for all nodes. Furthermore, $|x - y|$ denotes the Euclidean distance between two nodes x and y. Let $N_{1,i}$ denote i's one-hop neighbors, i.e. all nodes in immediate communication range of i, with $N_{1,i} = \{j : |j - i| \leq R;\ \forall\, j \in N\}$. Let σ_i denote the number of i's one-hop neighbors, i.e. $\sigma_i = |N_{1,i}|$. For an easier representation and without loss of generality, we denote i's one-hop neighbors as $j \in N_{1,i} = \{1, \ldots, \sigma_i\}$, where i itself is part of $N_{1,i}$. Let m denote the propagation range of messages, i.e. messages sent by a node i reach all nodes in the network that can be reached by at most m hops (also called m-hop neighborhood).

3.4. Create Key Revocation Lists (KRLs)

Each node i creates a key revocation list $\mathcal{KRL}^i(t_x)$ for all its known nodes $j \in N_i$. For an easier representation and without the loss of generality, we assume that $N_i = \{ID_1, \ldots, ID_{\Omega_i}\}$. A key revocation list $\mathcal{KRL}^i(t_x)$ can be represented as (Ω_i x ($\Omega_i + 3$))-matrix as shown below

$$\mathcal{KRL}^i(t_x) = \begin{pmatrix} a^i_{1,1} & \cdots & a^i_{1,j} & \cdots & a^i_{1,\Omega_i} & ID_1 & v^i_1 & X^i_1 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ a^i_{j,1} & \cdots & a^i_{j,j} & \cdots & a^i_{j,\Omega_i} & ID_j & v^i_j & X^i_j \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ a^i_{\Omega_i,1} & \cdots & a^i_{\Omega_i,j} & \cdots & a^i_{\Omega_i,\Omega_i} & ID_{\Omega_i} & v^i_{\Omega_i} & X^i_{\Omega_i} \end{pmatrix}, \qquad (3)$$

in which accusation values are represented as $a^i_{k,j} \in \{0, 1\}$ with $\{k, j\} \in \{1, \ldots, \Omega_i\}$. Note that $\mathcal{KRL}^i(t_x)$ must be adjusted every time Ω_i changes, i.e. a new column must be inserted into the matrix for each new node in N_i. Value $a^i_{k,j}$ indicates that node i "heard" that node j either accuses node k of malicious behavior ($a^i_{k,j} = 1$) or believes k is trustworthy ($a^i_{k,j} = 0$). The upper index i denotes that values are current values in i's $\mathcal{KRL}^i$. Note that other nodes l might have different values stored in their revocation lists $\mathcal{KRL}^l$, e.g. $a^i_{k,j} \neq a^l_{k,j}$ for $i \neq l$ in some cases. Discrepancies in accusation

values may exist, because accusations take different paths and times to propagate through the network. In the remainder, we use $\mathcal{KRL}^i$ for short because we only consider the current expiry interval.

Each j-th column vector in $\mathcal{KRL}^i$ for $1 \leq j \leq \Omega_i$, short $\underline{c}_j^i$, contains all accusations $a_{k,j}^i$ made by node j against nodes $k \in N_i$. Each j-th row vector in $\mathcal{KRL}^i$ for $1 \leq j \leq \Omega_i$, short $\underline{r}_j^i$, corresponds to a node $j \in N_i$ and contains, among other information, the accusation values $a_{j,k}^i$ from all nodes $k \in N_i$ evaluating node j. In particular, elements 1 to Ω_i in $\underline{r}_j^i$ contain accusation values $a_{j,1}^i$ to a_{j,Ω_i}^i. Element $(\Omega_i + 1)$ contains the identity ID_j of node j, element $(\Omega_i + 2)$ the current version number v_j^i of public key $Q_j(t_x)$, and the last element $(\Omega_i + 3)$ contains a 1-bit flag X_j^i that, when set, indicates that node i considers public key Q_j of node j as revoked. Node i sets

$$X_j^i = \begin{cases} 1 & \text{if} \quad a_{j,i}^i = 1 & \text{(Condition 1)} \\ & \text{or if} \quad a_{j,j}^i = 1 & \text{(Condition 2)} \\ & \text{or if} \quad \sum_k a_{j,k}^i \geq \delta \;\forall\; k \in \Omega_i \text{ with } X_k^i = 0 & \text{(Condition 3)} \\ 0 & \text{else} \end{cases} \tag{4}$$

Basically, node i considers j's public key as revoked, i.e. $X_j^i = 1$, if at least one of Conditions 1-3 is true. Condition 1 describes the case that node i observed the malicious behavior of node j during its own neighborhood watch (see Section 4.1.1). Condition 2 covers the case that i received a harakiri message from j indicating its own private key d_j has been compromised (see Section 4.1.2). And finally, Condition 3 considers the case in which node i received at least δ accusations against node j from nodes $k \in \Omega_i$ with $X_k^i = 0$. If none of the three conditions applies, node i considers node j and its current public key $Q_j(t_x, v_j^i)$ as trustworthy, i.e. $X_j^i = 0$.

When first creating its key revocation list $\mathcal{KRL}^i$, node i initializes all accusation values with $a_{k,j}^i = 0$ for all $\{k, j\} \subset \{1, \ldots, \Omega_i\}$. As a consequence, all revocation values $X_j^i = 0$. We assume fixed expiry intervals with $t_{x+1} = t_x + \Delta T$, i.e. all values in $\mathcal{KRL}(t_x)$ are re-set every ΔT. Once an accusation value $a_{k,j}^i$ is set (i.e. $a_{k,j}^i = 1$), the value will not be reset to zero until a new public key $Q_k(v', t_x)$ with $v' > v_j^i$ is received or a new time interval $t_x' > t_x$ starts.

3.5. Trust Model

First of all, we assume that the external KGC is honest, not compromised and trusted by all nodes. In applications where this is difficult to ensure, a

distributed KGC can be deployed [2, 7, 13]. Furthermore, the KGC checks the identities of nodes before they receive their private keys.

We define malicious nodes as nodes that are either compromised or selfish. We assume that compromised nodes will engage in some kind of malicious activities, otherwise these nodes cannot be detected. Selfish nodes, on the other hand, rather save their energy than forwarding other nodes' packets.

The accuracy of our revocation scheme depends on the employed monitoring scheme, i.e. false positive rate α and false negative rate β. In particular, α is the ratio of falsely accused nodes to all honest nodes and β is the ratio of undetected malicious nodes to all malicious nodes. Hence, $0 \leq \alpha, \beta \leq 1$, with typical values ranging from $0.01 - 0.1$.

We need the following definitions before we can derive the trust model:

Definition 2 (Direct Accusations). *Accusations received from one-hop neighbors containing the result of their neighborhood watch.*

For example, node i receives column vector $\underline{c}_j^j$ from an one-hop neighbor $j \in N_{1,i}$.

Definition 3 (Reported Accusations). *Accusations received from one-hop neighbors reporting accusations from nodes in l-hop distance, with $l > 1$.*

For example, node i receives column vectors $\underline{c}_k^j$ from an one-hop neighbor $j \in N_{1,i}$, where $k \in N \setminus N_{1,i}$.

Definition 4 (Trusted Node). *Node i trusts all nodes $j \in N_i$ with $X_j^i = 0$.*

Note that "trust" is not universal in our scheme, trust has to be put in a relation. For instance, node i trusts node j, whereas another node k might not trust j.

We are now ready to derive the trust model for our revocation scheme, in which each node i:

1. trusts that one-hop neighbors $j \in N_{1,i}$ that have been identified as malicious in i's neighborhood watch (with $a_{j,i}^i = 1$) are indeed malicious.
2. accepts direct accusations of any trusted one-hop neighbor $j \in N_{1,i}$.
3. accepts the majority vote of reported accusations from a group of at least ε trusted one-hop neighbors.

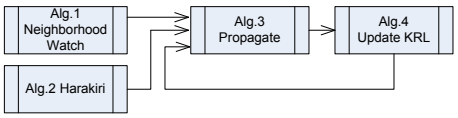

Figure 1: Overview Revocation Scheme

4. trusts δ or more accepted accusations (both direct and reported) against a node $j \in N_i$ to justify the revocation of j's keys.

From the first trust assumption it follows that $a^i_{j,i} = 1$ leads to $X^i_j = 1$. On the other hand, trust in a node follows that direct and reported accusations are accepted and counted towards the δ revocation threshold. Security parameter ε ensures that reported accusations have been observed by a group of trusted one-hop neighbors.

There are some universal bounds for the security parameters, namely $1 \leq \varepsilon \leq \sigma_i$ and $1 \leq \delta \leq \Omega$. We derive tighter bounds in Section 5.

4. Key Revocation Scheme

In this section, we first present a modified version of the revocation scheme in [11] that: 1) removes the requirement for knowing all nodes in m-hop distance while maintaining an m-hop propagation range; 2) is resilient to replay attacks; 3) prevents attacks on the hopcount of harakiri messages by ensuring network wide distribution; 4) increases resilience to battery exhaustion attacks, 5) has binary accusation values. We refer to this scheme as *basic revocation scheme*. Then, in Section 4.2, we show various ways to extend the basic scheme such that the resulting schemes can provide more pertinent security functionalities for different applications.

4.1. Basic Revocation Scheme

The basic revocation scheme consists of four algorithms: *Algorithm 1: Neighborhood watch*, *Algorithm 2: Harakiri*, *Algorithm 3: Propagate*, and *Algorithm 4: Update KRL*. Algorithms 1, 2 and 4 each require the propagation of the respective messages, respectively and thus all trigger Algorithm 3. An overview of the key revocation scheme is depicted in Figure 1.

4.1.1. Algorithm 1: Neighborhood Watch

The neighborhood watch algorithm is a local monitoring scheme, in which each node i monitors all neighbors in its one-hop neighborhood $N_{i,1}$. Whenever i observes a suspicious neighbor $j \in N_{1,i}$, i sets $a^i_{j,i} = 1$. Every time node i changes at least one accusation value $a^i_{j,i}$ from 0 to 1, i.e. from trusted to malicious status, i creates an neighborhood watch message $nm_{i,j}$ for each one-hop neighbor $j \in N_{1,i}$ according to

$$nm_{i,j} = (f_{K_{i,j}}(ID_i, nm_i, hopcount), (ID_i, nm_i, hopcount)), \forall\ j \in N_{1,i}. \quad (5)$$

A neighborhood watch message $nm_{i,j}$ contains the identity of the sender, here ID_i, and the observations from i's neighborhood watch denoted as nm_i. For simplicity, we choose $nm_i = \mathcal{KRL}^i$, i.e. i submits its entire key revocation list. More bandwidth efficient solutions can be designed. The last field is the *hopcount* that ensures that the message reaches all nodes in m-hop distance. Therefore, node i initially sets $hopcount = m$. To avoid unauthorized or modified accusations, accusation messages are protected by a MAC function $f()$, where pairwise pre-shared keys $K_{i,j}$ from (2) serve as MAC keys. After computing all accusation messages, node i starts Algorithm 3 to propagate them.

4.1.2. Algorithm 2: Harakiri

When a node i realizes that its private key d_i has been compromised, i creates a harakiri message hm_i with

$$hm_i = (ID_i, d_i, Q_i, (t_x, v_i), \text{“revoke”}). \quad (6)$$

The message contains the sender's identity (ID_i), the compromised private key d_i, the corresponding public key Q_i, the expiry date and version number of the public key (t_x, v_i), and a text string that marks the message as revocation message. Upon creating harakiri message hm_i, i starts Algorithm 3.

4.1.3. Algorithm 3: Propagate

In this algorithm nodes securely propagate accusations to their one-hop neighbors. Accusation messages am_i can be neighborhood watch messages $nm_{i,j}$ (see (5)), harakiri messages hm_i (see (6)), or update messages $um_{i,j}$ (see (9)), i.e. $am_i \in \{nm_{i,j}, hm_i, um_{i,j}\}$. The initiator of Algorithm 3, i.e. sender i sends its accusation message(s) to all its one-hop neighbors $j \in N_{1,i}$.

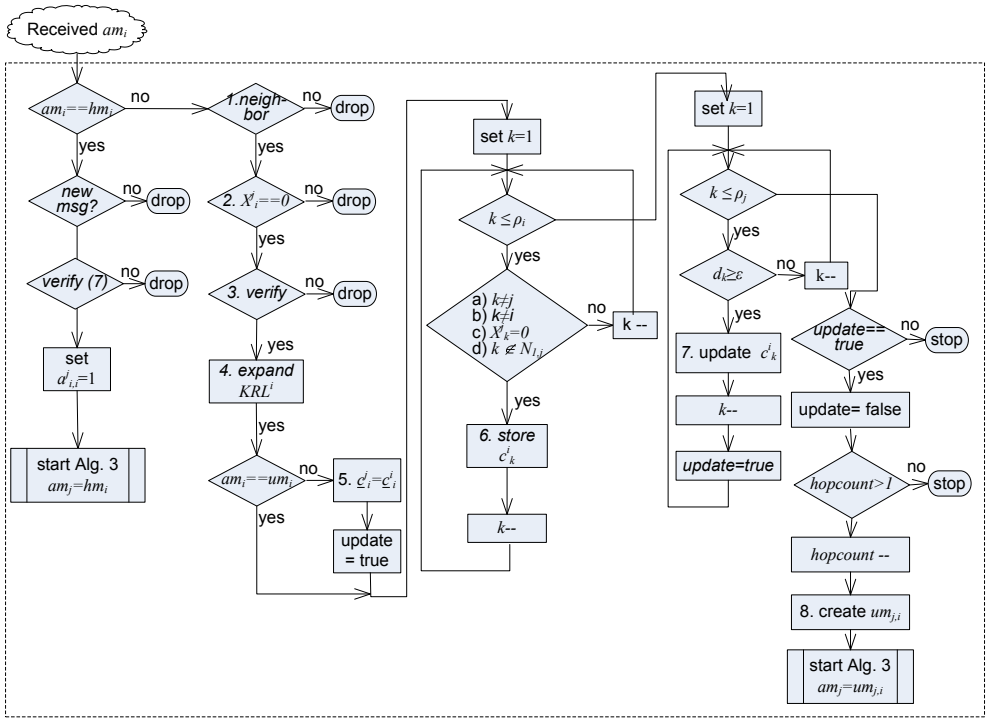

Figure 2: Flowchart Revocation Algorithm 4 *Update KRL*

4.1.4. Algorithm 4: Update KRL

In this algorithm, node i updates its key revocation list $\mathcal{KRL}^i$ using the received accusation messages am_j. We distinguish between three types of updates according to the received message am_j and describe each update process in the following paragraphs. The algorithm with all its processing steps is illustrated in Figure 2.

Received $am_j = hm_j$. First, receiver i checks wether it has already received that harakiri message. For example, this can be done by storing the hash values of the last ϖ received harakiri messages. If the same message has been previously received, i discards hm_j and aborts the algorithm. Else, i first needs to verify whether the message is authentic. It is not necessary to check who sent the message, because in any case the public key that corresponds to the broadcasted private key d_j should not be used any longer. However, it needs to be verified whether the broadcasted private key d_j corresponds to public key Q_j and identity ID_j. Therefore, a node i verifies

whether

$$K_{i,j} = \hat{e}(d_j, Q_i) \tag{7}$$

is true. If i is neither in possession of $K_{i,j}$ nor Q_j, i first computes Q_j from the received ID_j, t_j and v_j according to (1) and checks whether the received ID_j and the computed Q_j correspond to each other. If this check is successful, i derives $K_{i,j}$ according to (2). Finally, i checks whether (7) is true and if successful, i updates its key revocation list $\mathcal{KRL}^i$ by setting accusation value $a^i_{j,j} = 1$. Hence, Condition 1 in (4) is satisfied and i sets $X^i_j = 1$. Finally, node i starts Algorithm 3 with $am_i = hm_j$.

Received $am_j = nm_{j,i}$. When node i receives a neighborhood watch message $nm_{j,i}$, it needs to execute several verification and update steps. If a step is successful, i continues with the next step, else i drops the packet and aborts the algorithm. Upon receiving $am_j = nm_{j,i}$, node i performs the following steps:

1. *neighbor check*: i checks whether sender j is a direct neighbor, i.e. $j \in N_{1,i}$.
2. *check trustworthiness*: i checks whether it trusts j, i.e. $X^i_j = 0$.
3. *verify message authenticity*: i verifies the MAC of the received message $nm_{j,i}$ using pre-shared key $K_{i,j}$.
4. *expand key revocation list*: Node i scans $\mathcal{KRL}^j$ for rows of nodes $k \notin N_i$ and for any such k creates a new row $\underline{r}^i_k$ in $\mathcal{KRL}^i$, sets $N_i := N_i + ID_k$ and $\Omega_i := \Omega_i + 1$.
5. *copy direct accusations*: i extracts column vector $\underline{c}^j_j$ from nm_j to update its own column vector $\underline{c}^i_j$ in $\mathcal{KRL}^i$. For the update, node i only copies accusation values $a^j_{k,j} = 1$ to allow only transitions from 0 to 1. Upon completion, node i sets the update flag, i.e. $update = true$.
6. *store reported accusations*: i scans through all columns $\underline{c}^j_k$ with $k \in \{1, \ldots, \Omega_j\}$ in $\mathcal{KRL}^j$ and stores all columns $\underline{c}^j_k$ for which *all* following conditions hold:
   (a) $k \neq i$
   (b) $k \neq j$
   (c) $X^i_k = 0$
   (d) $k \notin N_{1,i}$

   Node i checks Conditions (a)-(d) for all $k \in \{1, \ldots, \Omega_j\}$. If all conditions are met for k, i stores $\underline{c}^j_k$ and increments counter r_k. All other

columns are discarded. We refer to the stored vectors as k-vectors. As discussed in our trust model in Section 3.5, we need a minimum of ε received k-vectors from different one-hop neighbors to establish trust in the reported accusations of node k. If at least ε k-vectors are collected, i.e. $r_k \geq \varepsilon$, i updates its $\mathcal{KRL}^i$ as described in the next step.

7. *use accumulated k-vectors for update*: Node i checks for all $k \in \{1, \ldots, \Omega_i\}$ whether $r_k \geq \varepsilon$. Whenever true node i updates the k-vector in $\mathcal{KRL}^i$. For an easier representation and without loss of generality, we assume i stored r_k column vectors $\underline{c}_k^j$ from r_k one-hop neighbors j with $j \in \{1, \ldots, r_k\}$ in Step 6, with $r_k \geq \varepsilon$. Each accusation value $a_{l,k}^i$ with $l \in \{1, \ldots, \Omega_i\}$ in $\underline{c}_k^i$ is computed from the majority vote over all collected $a_{l,k}^j$, with

$$a_{l,k}^i = \begin{cases} 1 & \text{if } \sum_{j=1}^{r_k} a_{l,k}^j > \frac{r_k}{2} \\ a_{l,k}^i & \text{else} \end{cases}, \tag{8}$$

   where $j \in \{1, \ldots, r_k\}$. Basically, if the majority of the accumulated accusation values $a_{l,k}^j$ against a node l equal 1, node i sets the value to 1. Otherwise, the accusation value in $\mathcal{KRL}^i$ remains unchanged. This ensures that only transitions from 0 to 1 are allowed. Note that all k-vectors that have been used for the KRL update are erased, whereas "unused" k-vectors with $r_k < \varepsilon$ remain in i's storage. If i updated at least one k-vector in $\mathcal{KRL}^i$, node i sets the update flag $update = true$.
8. *prepare update message*: If $update = true$, which is always true for $am_i = nm_{i,j}$, node i prepares an update message $um_{i,j}$ for all its one-hop neighbors $j \in N_{1,i}$ with

$$um_{i,j} = (f_{K_{i,j}}(ID_i, um_i, hopcount), (ID_i, um_i, hopcount)), \forall\ j \in N_{1,i}. \tag{9}$$

   The messages are constructed similar to the neighborhood watch messages $nm_{i,j}$ in (5). For simplicity, we assume $um_i = \mathcal{KRL}^i$, where more bandwidth efficient solutions such as only sending updated vectors are possible. All messages $um_{i,j}$ for all $j \in N_{1,i}$ serve as input to Algorithm 3 and thus are propagated to i's one-hop neighborhood. After triggering Algorithm 3, the update flag is reset, i.e. $update = false$.

Received $am_j = um_{j,i}$. When node i receives a KRL update message $am_j = um_{i,j}$ from j, node i executes Steps 1-4 and 6-7 as described in the

previous paragraph for $am_j = um_{i,j}$. If node i updated at least one of its k-vectors, i.e. $update = true$ and $hopcount > 1$, i creates an update messages $um_{i,j}$ according to (9) for all $j \in N_{1,i}$ with $hopcount := hopcount - 1$, starts Algorithm 3, and resets the update flag.

4.2. Extensions of the Basic Revocation Scheme

In this section, we describe two classes of extensions to the presented basic revocation scheme.

4.2.1. Key Escrow Prevention and Privacy Protection

The first class of extensions deal deals with the inherent key escrow property of IBC schemes and other privacy related issues.

1. *Distributed On-line KGC.* Our scheme can be easily modified for MANETs with distributed on-line KGCs, because the revocation scheme remains unchanged while the distributed on-line KGC takes over the task of key generation and distribution. For example, (k, n)-threshold schemes can be used to distribute master key s to all network nodes such that k D-KGCs can collaboratively generate and distribute (new) private keys, as in [7, 13, 14, 20, 21].
2. *Confidential Accusations.* To avoid that (malicious) nodes can overhear accusations, accusations can be encrypted, as suggested in [20]. For accusation confidentiality, any symmetric encryption algorithm can be used with a secret encryption key $K'_{i,j}$ that is derived from the pre-shared keys $K_{i,j}$ in (2). In that case, all neighborhood watch and update messages are encrypted and only sent to neighbors that are not accused in these messages, e.g. $nm_{i,j} = (f_{K_{i,j}}(ID_i, nm_i), E_{K'_{i,j}}(ID_i, nm_i)), \forall\, j \in N_{1,i} \setminus \{L\}$, where L is the set of accused neighbors.
3. *Dedicated Key Pairs.* Harakiri messages hm_i (see (6)) contain private keys d_i which affects the security of all previous messages that were either signed under d_i or encrypted under Q_i or $K_{i,j}$ for any $j \in N$. To prevent misuses of self-revoked keys, we suggest using dedicated private and public key pairs for different purposes. For example, public keys could contain a label that specifies the purpose of the keys, e.g. $Q_i(t_x, v_i) = H_1(ID_i||t_x||v_i||\text{label})$, where label $\in \{$ sign, encrypt, revocation$\}$. This format still allows nodes i to derive public keys Q_j and pre-shared keys $K_{i,j}$ of other nodes j in a non-interactive fashion while particular private keys can be revealed without revealing private keys dedicated to other purposes.

4. *Crypto Agility.* The presented key revocation scheme can be adopted to other cryptographic schemes, such as PKI and secret key based security solutions, by adapting the respective system set up accordingly. The actual revocation scheme algorithms 1-4 as well as security parameters and the underlying trust model would remain the same.

4.2.2. Adaptive and Versatile Schemes

The second class of extensions further enhance the adaptability and versatility of the basic scheme.

1. *Regaining Trust.* After a previously marked malicious node j behaves well for a certain period of time T_R, its neighbors i may update the corresponding accusation values accordingly, e.g. reducing accusation value $a^i_{j,i}$ by $\Delta a \leq 1$. Note that this works better when accusation values are real numbers.
2. *Sleeping nodes.* Sleeping cycles of nodes can be easily integrated into the revocation scheme. Before going to sleep, nodes must inform their neighbors to avoid being marked as malicious. After waking up, a node can instantly continue its neighborhood watch and additionally request updates from its one-hop neighbors.
3. *Adapting Scheme to Hostile Environments.* The presented key revocation scheme can be adapted to many different environments. For instance in very hostile environments, all accusation values could be initialized with ones, the majority vote could be computed differently, and updates triggered by different events. These parameters should be selected according to the fraction of expected malicious nodes.
4. *Adaptive Monitoring Schemes and Security Parameters.* Sometimes it might be advisable to adjust the system parameters in the running system. For instance, some nodes might have a number of accusations always just below threshold δ. In that case the threshold of these or all nodes should be dropped accordingly at the next expiry interval. In other cases it might be advisable to have an adaptive monitoring scheme, e.g. some nodes might be at the edge of the network and do not need to forward many packets. In that case, a monitoring scheme that only monitors the number of forwarded packets is not sufficient. It is advisable to have an adaptive monitoring scheme that implements several kinds of metrics and thresholds that can be selected according to some network parameters, such as network density, number of neighbors, position in the network, etc..

5. *Network-wide Revocations.* The propagation range m can be removed such that accusations are propagated throughout the entire network, as it is done for harakiri messages. The disadvantages of this modification are increased communication load and storage requirements. Hence, m serves as performance parameter that should be selected according to the number of malicious nodes, the mobility of malicious nodes, available network bandwidth and power constraints of nodes.
6. *Adversary Models.* We treat the implemented monitoring scheme as a black box with false positive and false negative rates α and β, respectively and specify neither how malicious behavior is defined nor how such behavior can be measured. This makes our revocation scheme very versatile and suitable to thwart all types of existing as well as potential future attacks. In order to thwart specific malicious behavior, the presented revocation scheme does not need to be modified and only the monitoring scheme must be modeled accordingly. For instance, if the adversary model describes DoS attacks, the monitoring scheme could be set to detect large numbers of sent packets. If the adversary model describes blackhole attacks, the monitoring scheme could be set to check the number of dropped packets.

5. Security Analysis of Attacks by Outsiders and Non-colluding Insiders

We assume that the underlying IBC scheme including the pre-shared keys from (2) is secure and we limit our security analysis to the proposed key revocation scheme.

5.1. Outsider Attacks

Message Fabrications and Modifications: In the revocation scheme, all neighborhood watch and update messages are protected with pre-shared keys $K_{i,j}$. Thus the messages provide message authentication and integrity protection and can be neither fabricated nor modified by outsiders. On the other hand, harakiri messages are not protected but contain private and public key pairs. Hence, harakiri messages can only be created by insiders or adversaries who compromised a node. Latter have no reason to send harakiri messages, because it would render the key compromise useless.

Replay Attacks: An outsider may replay previously observed messages. Note that receiving multiple copies of a message can also be caused by the

multi-path propagations in MANETs, which may be considered as unintentional replay attack.

Old public keys cannot be successfully replayed in the presented revocation scheme because they contain an expiry date and a version number and thus cannot be used to replace newer keys. For the same reason, messages can only be replayed in the same expiry interval (i.e. for $(t_x - \Delta T) \leq t \leq t_x$), because neighborhood watch, harakiri and update messages all contain keys with expiry date t_x. Now recall that accusation and thus revocation values can only be changed from 0 to 1 in a key revocation list unless a new key is received or a new expiry interval starts, and each node has only one vote towards the δ threshold. Hence, replayed harakiri messages, replayed accusation messages containing more accusation values set to zero than in the current $\mathcal{KRL}^i$, as well as accusation values containing the same number of values set to one, will be all discarded. On the other hand, replayed accusations cannot contain more ones, because such messages could have not been previously observed by an adversary. However, replayed reported accusations that have been previously counted towards the majority vote, may be counted again as part of a new majority vote computation. However, the majority vote only allows changes from 0 to 1 (see (8)) and is furthermore protected by security parameter ε, i.e. a replayed message still only counts $1/\varepsilon$ towards one accusation value.

A similar discussion applies to messages that arrive out of order (maliciously or unintentionally). For example, if first an accusation value set to one is received and later an older message containing the same accusation value set to zero, the second received message is discarded.

Battery Exhaustion: An outsider could attempt draining a node's battery in a so-called battery exhaustion attack [18] by repeatedly sending messages that cause a node to perform demanding computational operations. The impact of these attacks on our revocation scheme is limited, because replayed harakiri messages will be discarded and nodes only accept accusation messages from trusted one-hop neighbors. These checks are very efficient (a table look up) and thus cannot be exploited to drain a battery. If an adversary spoofs the identity of a trusted one-hop neighbor, the attack would be detected when verifying the authenticity of the first bogus message. Here, the verification costs are rather small for neighborhood watch and update messages (namely a MAC computation) and more demanding for verifying harakiri messages (up to two pairing computations). The attack cannot be successfully continued if the monitoring scheme is set to mark the sender's

identity as malicious after several failed attempts.

Compromised Nodes: Under the assumption that the underlying cryptographic scheme is secure, an outsider adversary can only obtain keying material by compromising a node. The adversary can use the compromised keys until they are revoked or expired because outsiders cannot request new keys for a compromised node. Hence, an adversary can only launch an attack from the time a node is compromised t_c until the key is revoked at t_r, i.e. during a time interval $\Delta T_{attack} = t_r - t_c$, where $0 \leq \Delta T_{attack} \leq \Delta T$. In the remaining security analysis we show how the security parameters of our revocation scheme should be selected to minimize the time it takes to detect a malicious node and revoke its keys. Most importantly, a single node cannot force false accusations due to security parameters δ and ε. We will discuss this in detail in the next section for non-colluding insiders.

5.2. Non-colluding Insiders

Sybil Attacks: Malicious nodes could try to bypass security parameter δ by fabricating δ different identities in a so-called Sybil attack [8]. Our scheme uses ID-based public keys of a fixed format, i.e. upon identifying to the KGC, a node can only obtain one possible valid private key for a specific expiry date. Hence, Sybil attacks requiring keying material are not applicable in our scheme. In another type of Sybil attack that does not require keying material, an adversary j creates δ rows in its key revocation list for δ virtual nodes $V = \{v_1, v_2, \ldots, v_\delta\}$ and then sets $a^j_{i,x} = 1$ for all $x \in V$. However, the reported accusations received from j are only accepted by a node i, if at least $\varepsilon - 1$ other reported accusations from the same δ nodes in S are received. Hence, the attack requires at least ε colluding one-hop neighbors and will be covered in Section 6.

In another Sybil attack described in [17], an adversary replicates nodes. However, such adversaries do not gain any advantage in our revocation scheme, because only one accusation is counted for each node with the same identity.

Malicious Nodes: A malicious node k could attempt to impersonate another node i. However, k's keying material cannot be used to obtain i's keying material. Hence, the only way to impersonate another node i is compromising this node (see previous section).

An undetected malicious node attempting to launch a battery exhaustion attack, could send messages that would be initially accepted because they pass the verifications. However, eventually the attack would be detected.

A single malicious node k may send more than one accusation against the same node j, however each receiver i only stores one accusation value $a^i_{j,k}$. In other words, each node has only one vote towards the δ threshold.

Malicious nodes cannot simply drop accusations against themselves, because this will be detected in the neighborhood watch scheme. Besides, accusations are broadcasted and, thus, still reach other nodes, even if one of the propagation paths is broken. Attempts of malicious nodes to modify accusations against themselves are prevented by using integrity protected accusations. An adversary could modify its own key revocation list. However, the impact of this attack is limited by security parameters δ and ε. In particular, a malicious node's false direct accusations count only $1/\delta$ towards the δ-threshold for a revocation; while δ false reported accusations count only $1/\varepsilon$ towards the ε-threshold of reported accusations. Hence without collusion a malicious node cannot force the revocation of a node.

Roaming Adversaries: Whenever a node's accusation count approaches δ, that node may move to a new neighborhood, say at least $2m$ hops away, hoping that nodes there have not received the accusations against it. We refer to these kind of malicious nodes as *roaming adversaries*. A roaming adversary k's new one-hop neighbors will eventually detect k's malicious behavior and thus k needs to move again before its key is revoked. Lets assume nodes are uniformly distributed and each routing hop is over a distance R. Then, the speed S that is necessary for roaming adversaries to travel to a new neighborhood in 2 hop distance before their current keys expire at time t_x is

$$S \geq \frac{2mR}{t_x - t}.$$

Note that t is the current time and thus $t_x - t \leq \Delta T$. We can observe that by selecting m sufficiently large and the expiry intervals ΔT small, roaming adversaries have to travel fast to escape the revocation of their keys. Due to the dynamics of MANETs, moving $2m$ hops does not guarantee that nodes in the new neighborhood have not received some previous accusations. Hence, adversaries cannot remain at the same location for a long period of time and need to travel faster each time they change the neighborhood with t advancing t_x. Finally, the threat posed by mobile adversaries can be completely thwarted by encrypting all accusations, as described in Section 4.2, because adversaries would not learn about the number of accusations against them.

Falsely Accused Nodes: For the remainder of the security analysis, we assume that all nodes implement the same monitoring scheme and introduce

Table 2: List of Notations for Security Analysis

H_i, $n_h^i = \|H_i\|$	Set & number of i's honest one-hop neighbors
F_i, $n_f^i = \|F_i\|$	Set & number of i's falsely marked honest one-hop neighbors
M_i, $n_m^i = \|M_i\|$	Set & number of i's malicious one-hop neighbors
U_i, $n_u^i = \|U_i\|$	Set & number of i's malicious undetected one-hop neighbors
C, $n_c = \|C\|$	Set & number of colluding nodes
Θ_i	Set of i's trusted one-hop neighbors

some notations, summarized in Table 2. Lets H_i denote i's honest one-hop neighbors, with $|H_i| = n_h^i$, and M_i i's malicious one-hop neighbors, with $|M_i| = n_m^i$. That follows that $N_{1,i} = H_i \bigcup M_i$, where $H_i \bigcap M_i = \emptyset$ and thus $\sigma_i = n_h^i + n_m^i$. Furthermore, F_i denotes i's honest one-hop neighbors that have been falsely marked as malicious by i, with $|F_i| = n_f^i$, and U_i denotes i's undetected malicious one-hop neighbors, with $|U_i| = n_u^i$. Hence, $F_i \subseteq H_i$ and $U_i \subseteq M_i$. Finally, the colluding nodes are denoted as C with $|C| = n_c$. In our analysis of colluding one-hop neighbors, we consider the case that all undetected malicious nodes collude, i.e. $C = U_i$ and $n_c = n_u^i$.

We can observe that false positive rate α causes a node to falsely mark $n_f^i = \alpha n_h^i$ of its honest one-hop neighbors as malicious. Note that falsely accused nodes do not directly pose a security threat. However, besides the inconvenience false accusations may cause the accused nodes, a large number of falsely accused nodes could stop the revocation scheme from working efficiently or in the worst case from working at all. To be able to revoke keys, a node i must receive at least δ (direct and/or reported) accusations from trusted nodes. Each node i trusts all its one-hop neighbors $n \in \Theta_i = (H_i \setminus F_i) \bigcup U_i$, which follows that the number of trusted one-hop neighbors is $|\Theta_i| = (1-\alpha)n_h^i + \beta n_m^i$. To enable key revocations by exclusively direct accusations $|\Theta_i| \geq \delta$ must hold. However, to enable the acceptance of reported accusations (which is necessary to revoke keys from nodes that are more than 2 hops away) and in the case that $\delta > \sigma_i$, the false positive rate α of a revocation scheme must satisfy

$$\frac{n_h^i + \beta n_m^i - \varepsilon}{n_h^i} \geq \alpha.$$

6. Security Analysis of Attacks by Colluding Nodes

We now analysis the resilience of the scheme to colluding nodes. In particular, we show how security parameters δ and ε, as well as false negative rate β should be chosen to prevent such attacks.

6.1. Colluding One-hop Neighbors

Definition 5 (Successful Attack by Colluding One-hop Neighbors). *A group of $n_c \leq \sigma_i$ colluding one-hop neighbors can convince an honest node i to mark the key of another honest node $j \in N$ as revoked in $\mathcal{KRL}^i$, i.e. $X_j^i = 1$.*

A monitoring scheme with false negative rate β leads to $n_u^i = \beta n_m^i$ undetected malicious nodes in i's one-hop neighborhood. Hence, up to n_u^i one-hop neighbors $u \in U_i$ may collude to launch an attack. Colluding one-hop neighbors u can launch two types of attacks: *A. Altering direct accusations*, i.e. nodes u alter their own accusations as part of their propagated neighborhood watch messages, and *B. Altering reported accusations*, i.e. nodes u alter reported accusations of nodes that are 2 or more hops away.

Altering direct accusations: Recall that δ accusations revoke a key (see Condition 3 in (4)) and that node i copies the direct accusations of all trusted one-hop neighbors $j \in \Theta_i$ (see Step 5 in Algorithm 4). We now consider the following attack by colluding nodes $u \in U_i$:

- each node $u \in U_i$ sets $a_{j,u}^u = 1$ for an honest node $j \in N$ and sends a neighborhood watch message

Upon receiving $nm_{u,i}$, i uses u's neighborhood watch vectors $\underline{c}_u^u$ to update $\underline{c}_u^i$ in $\mathcal{KRL}^i$. In that way, node i updates n_u^i vectors in its revocation list, each containing $a_{j,u}^i = 1$. Thus, there are at least n_u^i accusations against node j in $\mathcal{KRL}^i$. Hence, if the following inequality

$$n_u^i \geq \delta$$

holds, node i will revoke node j's key. This result is not surprising because δ is the threshold for our revocation scheme, and thus δ malicious undetected one-hop neighbors can revoke the key of any $j \in N$. The described attack can be prevented by selecting δ and β such that $n_u^i < \delta$. We know that

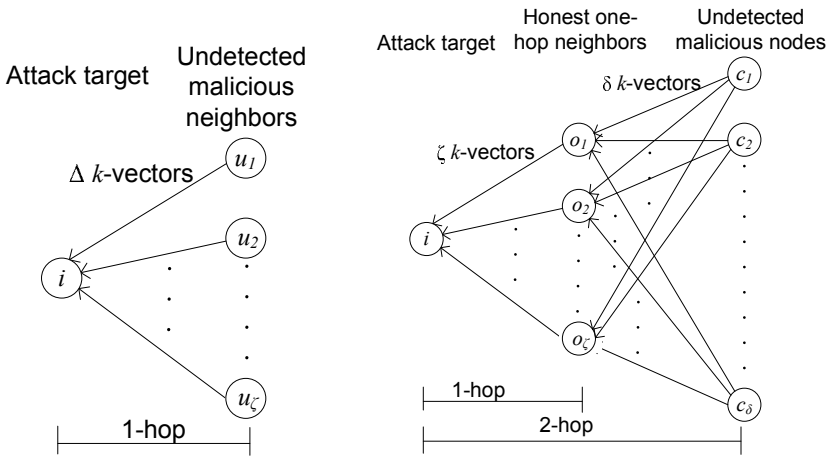

Figure 3: Attacks by Colluding Nodes: (a) One-hop Neighbors Altering Reported Accusations, (b) Two-hop Neighbors Altering Direct Accusations.

$n_m^i \leq \sigma_i$, thus attacks by altering direct accusations can only succeed if $\beta\sigma_i \geq \delta$. Hence, if we select β and δ such that

$$\beta < \frac{\delta}{\sigma_i}, \tag{10}$$

the described attack is completely prevented. We can observe that by selecting $\delta \geq \sigma_i$, (10) always holds because $\beta < 1$ for any monitoring scheme. In most monitoring schemes β typically ranges from 0.01 to 0.1 and thus δ can be selected smaller than σ_i, which does not put many restrictions on the parameter selection.

Altering reported accusations: In another attack by colluding one-hop neighbors $u \in U_i$, the adversaries alter reported accusations exploiting the majority rule (see (8)) that is applied by node i to derive column vectors $\underline{c}_k^i$, with $k \in N \setminus N_{1,i}$. From Steps 6 and 7 in Algorithm 4, we can observe that at least $\lfloor \frac{r_k}{2} + 1 \rfloor$ trusted nodes are necessary to gain majority and thus determine the accusation values in $\underline{c}_k^i$. In the attack, colluding nodes manipulate their submitted k-vectors. The colluding nodes $u \in U_i$ execute the following steps to launch an attack of type B:

- each $u \in U_i$ selects Δ nodes in $N \setminus N_{1,i}$, which is denoted as V, i.e. $V \subset N \setminus N_{1,i}$ and $\Delta = |V|$.

- node u sets $a^u_{j,v} = 1$ for all $v \in V$, and sends an update message $um_{u,i}$.

Upon receiving all n^i_u update messages $um_{u,i}$, node i updates the accusation value $a^i_{j,v}$ with $a^i_{j,v} = 1$ for all $v \in V$, if the received k-vectors from the colluders form the majority, i.e. if $n^i_u > \frac{r_k}{2}$ (see (8)). If the number of accusations reaches threshold δ, i revokes j's key with $X^i_j = 1$. Note that the minimum number of colluding nodes needed to force the acceptance of the reported accusations is $r_k = \varepsilon$. Hence the minimum number of colluding nodes $u \in U_i$ is $n^i_u = \lfloor \frac{\varepsilon}{2} + 1 \rfloor$. In the remainder of our security analysis we use $\zeta = \lfloor \frac{\varepsilon}{2} + 1 \rfloor$. The attack is illustrated in Fig. 3-(a) and we summarize the conditions for a successful Attack B below:

1. all $v \in V$ are trusted by node i, i.e. $X^i_v = 0$ for all $v \in V$
2. $\Delta \geq \delta$
3. $n^i_u \geq \zeta$

Condition 1 is a requirement for any group of colluding nodes and thus assumed to be true in this attack analysis. Condition 2 is also generally true because δ is typically chosen a lot smaller than $\Omega - \sigma_i$ and u can choose any node in more than 2 hop distance or even fabricate nodes (see Sybil attack). Therefore we focus on Condition 3. Recall that $n^i_m \leq \sigma_i$. Hence if we choose β and ε such that

$$\beta < \frac{1}{\sigma_i} \lfloor \frac{\varepsilon}{2} + 1 \rfloor, \tag{11}$$

the described attack is prevented. We would like to emphasize that (11) reflects the best possible scenario from the attackers' point of view, in which exactly $\lceil \frac{\varepsilon}{2} - 1 \rceil$ honest one-hop neighbors report k-vectors. Less or more honest nodes would both require a larger number of colluding nodes n^i_u, because $r_k < \varepsilon$ in the first case, whereas n^i_u must maintain the majority in the latter case.

Note that $1 \leq \varepsilon \leq \sigma_i$. Hence, selecting large ε relaxes the condition on the accuracy of the monitoring scheme while reducing the efficiency and functionality of our revocation scheme. However, in any case $\varepsilon \geq 1$ and thus selecting $\beta < \frac{1}{\sigma_i}$ ensures that Attack B is prevented for any selection of ε. Since σ_i varies for different neighborhoods an average value $\overline{\sigma}$ should be estimated for the network before selecting a monitoring scheme with appropriate β.

Remark 1. *Colluding one-hop neighbors can combine Attacks A and B, i.e. alter direct and reported accusations. In that case only $\Delta = \delta - n^i_u$ k-vectors*

must be manipulated, because the colluders send n_u^i altered direct accusations. Hence, the described attack modifies Conditions 1 and 2 but not Condition 3 and the attack can still be prevented by selecting β and ε such that (11) holds.

6.2. Colluding l-hop Neighbors

We now analyze attacks by colluding l-hop neighbors with $l > 1$.

Definition 6 (Successful Attack by Colluding l-hop Neighbors). *A group of $n_c \leq \Omega - \sigma_i$ colluding l-hop neighbors, with $l > 1$, can convince an honest node i to mark the key of another honest node $j \in N$ as revoked in $\mathcal{KRL}^i$, i.e. $X_j^i = 1$.*

Altering direct accusations: We first consider an attack by colluding two-hop neighbors, i.e. $l = 2$, in which the colluders alter their direct accusations. These altered accusations are received by the one-hop neighbors of the colluders, which in turn report the (altered) accusations to node i. We assume the following attacking scenario:

- a group of colluding 2-hop neighbors C, with $|C| = n_c$ and $C \subset (N_{2,i} \setminus N_{1,i})$
- a group of nodes O that are one-hop neighbors of node i as well as of all nodes $c \in C$, i.e. $O \subset N_{1,i}$ and $O \subset N_{1,c}$ for all $c \in C$

Furthermore, we assume that all nodes $o \in O$ faithfully execute the revocation algorithms. The attack consists of two phases:
Phase 1.

- each $c \in C$ sets $a_{j,c}^c = 1$ and sends a neighborhood watch message
- each receiver $o \in O$ updates its $\mathcal{KRL}^o$ with $a_{j,c}^o = 1$ if $X_c^o = 0$

Phase 2.

- each $o \in O$ sends an update message reporting the altered accusations, i.e. $a_{j,c}^o = 1$ for all $c \in C$
- node i updates its $\mathcal{KRL}^i$ if it received at least $r_c > \varepsilon$ c-vectors. In that case the majority vote forces i to set $a_{j,c}^i = 1$ for all $c \in C$. And if the number of collected accusations is larger than δ, i revokes j's key

The attack is illustrated in Fig. 3-(b) and works if the following three conditions hold:

1. $|C| \geq \delta$
2. $|O| \geq \zeta$
3. all $o \in O$ mark each $c \in C$ as honest, i.e. $X_c^o = 0$ for all $o \in O$ and $c \in C$

We assume the first two conditions to be true and analyze Condition 3, i.e. the probability that each colluder $c \in C$ remains undetected by the monitoring scheme of each of its one-hop neighbors $o \in O$. For an easier representation and without loss of generality, we denote the one-hop neighbors as $O = \{o_1, o_2, \ldots, o_\zeta\}$ and the colluding adversaries as $C = \{c_1, c_2, \ldots, c_\delta\}$. The probability that one adversary c_r with $r \in \{1, \ldots, \delta\}$ remains undetected by one neighbor o_s with $s \in \{1, \ldots, \zeta\}$ is

$$\frac{\beta n_m^{o_s}}{\sigma_{o_s}}.$$

The probability that c_r remains undetected by all considered ζ one-hop neighbors $o \in O$ is

$$\frac{\beta^\zeta n_m^{o_1} n_m^{o_2} \cdots n_m^{o_\zeta}}{\sigma_{o_1} \sigma_{o_2} \cdots \sigma_{o_\zeta}}.$$

Now each of the δ colluding attackers c_r must fool all one-hop neighbors $o \in O$, which leads to probability

$$\left(\frac{\beta^\zeta n_m^{o_1} n_m^{o_2} \cdots n_m^{o_\zeta}}{\sigma_{o_1} \sigma_{o_2} \cdots \sigma_{o_\zeta}} \right)^\delta.$$

Lets assume that all nodes o_s have approximately the same number of one-hop neighbors $\overline{\sigma}$ and approximately the same number of malicious one-hop neighbors $\overline{n}_m$, then the probability of a successful attack by δ colluding two-hop neighbors $c \in C$ is

$$\left(\frac{\beta \overline{n}_m}{\overline{\sigma}} \right)^{\delta\zeta}. \tag{12}$$

We know that $\overline{n}_m \leq \overline{\sigma}$ and $\beta < 1$, i.e. the term in brackets is smaller than 1. Furthermore, with typical values of β ranging between 0.01 up to 0.1, the probability of a successful attack becomes negligible for small β and larger exponents.

The described attack assumes that node i receives a total of ε c-vectors, where ζ contain the altered accusations. However, if node i does not receive any other c-vectors, the colluders must manipulate ε c-vectors and thus "convince" ε as opposed to ζ one-hop neighbors o, which further reduces the likelihood of the described attack.

Altering reported accusations: To increase their chance of a successful attack, the colluders in 2-hop distance could alter reported accusations of neighbors in $(l > 2)$-hop distance. In that attack, assuming the best possible case from the attackers' perspective, ζ instead of δ colluders are sufficient to launch the described attack. However, with

$$\left(\frac{\beta\overline{n}_m}{\overline{\sigma}}\right)^{\zeta^2} \tag{13}$$

the probability of a successful attack is only slightly larger and still negligible.

We now argue that the described attack for $l = 2$ is the best possible attack for colluding nodes in l-hop distance, with $l > 1$. Colluders must always fool at least ζ one-hop neighbors. Then in the best possible case (from the colluders' perspective), the altered accusations propagate through the network. Consequently, the probability of a successful attack by l-hop colluders can never exceed the probability in (13).

7. Discussions and Conclusions

In this article, we present a parameterized trust model and security analysis framework that can be applied to any monitoring-based scheme in MANETs or other decentralized networks. This work is the first to take the false positive and false negative error rates, α and β, of monitoring schemes into account and shows how they affect the functionality and security of monitoring-based schemes in MANETs. Our extensive security analysis considers a wide range of attacks. For example, we show how security thresholds δ and ε can be utilized to thwart Sybil and replay attacks while cryptographically protected messages can be used to prevent some other common attacks. We derive bounds and relations for security parameters δ and ε and system parameters α and β to counteract attacks by colluders. In addition, we show how the power of roaming adversaries can be restricted by choosing parameters ΔT and m accordingly or encrypting accusation messages.

We use these results to design a practical decentralized key revocation scheme that facilitates the detection of malicious nodes in MANETs and

revokes their keys. The introduced parameters and black-box approach of the monitoring scheme, make our revocation scheme scalable in security and performance as well as adaptable to the hostility of implementation environments and expected attacks. In addition, the presented revocation scheme algorithms can be seamlessly integrated into any pairing-based IBC security solution for MANETs and can be adapted to PKI and secret key schemes.

Acknowledgement

This work was supported by the Strategic Project Grant of NSERC, Canada.

References

[1] S. Bhargava and D.P. Agrawal. Security Enhancements in AODV Protocol for Wireless Ad Hoc Networks, *VTC 2001 Fall*, vol.4, pp. 2143-2147, 2001.

[2] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing, *Advances in Cryptology - CRYPTO '2001*, LNCS 2139, Springer Verlag, pp. 213-229, 2001.

[3] K. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R. Olsson. Detecting Disruptive Routers: A Distributed Network Monitoring Approach, *IEEE Symposium on Security and Privacy*, 1998.

[4] S. Buchegger and J. Boudec. Performance Analysis of the CONFIDANT Protocol (Cooperation of Nodes: Fairness in Dynamic Ad-hoc Networks), *MOBIHOC '02*, 2002.

[5] H. Chan, V. Gligor, A. Perrig, and G. Muralidharan. On the Distribution and Revocation of Cryptographic Keys in Sensor Networks. IEEE Transactions on Dependable and Secure Computing, vol. 2, no. 3, pp. 233 - 247, 2005.

[6] C. Crépeau and C.R. Davis. A Certificate Revocation Scheme for Wireless Ad Hoc Networks, *Proceedings of ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN '03)*, ACM Press, ISBN: 1-58113-783-4, pp. 54-61, 2003.

[7] H. Deng, A. Mukherjee, D.P. Agrawal. Threshold and Identity-based Key Management and Authentication for Wireless Ad Hoc Networks, *International Conference on Information Technology: Coding and Computing (ITCC'04)*, vol. 1, pp. 107-115, 2004.

[8] J. R. Douceur. The Sybil Attack, *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, LNCS 2429, Springer Verlag, pp. 251-260, 2002.

[9] X.X. Fan and G. Gong. Key Revocation based on Dirichlet Multinomial Model for Mobile Ad Hoc Networks, to appear in the proceedings of the *Fourth IEEE LCN Workshop on Network Security (WSN 2008)*, October 2008.

[10] K. Hoeper and G. Gong. Identity-Based Key Exchange Protocol for Ad Hoc Networks, Canadian Workshop of Information Theory CWIT '05, pp. 127-130, 2005.

[11] K. Hoeper and G. Gong. Key Revocation for Identity-Based Schemes in Mobile Ad Hoc Networks, *International Conference on AD-HOC Networks & Wireless (AD HOC NOW '06)*, LNCS 4104, Springer Verlag, pp. 224-237, 2006.

[12] J.P. Hubaux, L. Buttyán and S. Čapkun. The Quest for Security in Mobile Ad Hoc Networks, *ACM Symposium on Mobile Networking and Computing –MobiHOC 2001*, 2001.

[13] A. Khalili, J. Katz, and W. Arbaugh. Toward Secure Key Distribution in Truly Ad-Hoc Networks, *2003 Symposium on Applications and the Internet Workshops (SAINT 2003)*, IEEE Computer Society, ISBN: 0-7695-1873-7, pp. 342-346, 2003.

[14] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang. Self-Securing Ad Hoc Wireless Networks, *Seventh IEEE Symposium on Computers and Communications (ISCC '02)*, 2002.

[15] S. Marti, T. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehaviour in Mobile Ad Hoc Networks, *Sixth Annual International Conference on Mobile Commmunication and Networking*, 2000.

[16] T. Moore, J. Clulow, S. Nagaraja, and R. Anderson. New Strategies for Revocation in Ad-Hoc Networks. *Security and Privacy in Ad-hoc and Sensor Networks, 4th European Workshop, ESAS 2007*, LNCS 4572, pp. 232-246, 2007.

[17] B. Parno, A. Perrig, and V.D. Gligor. Distributed Detection of Node Replication Attacks in Sensor Networks. *2005 IEEE Symposium on Security and Privacy (S&P'05)*, IEEE Computer Society, pp. 49-63, 2005.

[18] F. Stajano and R. Anderson. The Resurrecting Duckling: Security Issues for Ad-Hoc Wireless Networks, *In Proceedings of the 7th International Workshop on Security Protocols*, LNCS 1796, Springer Verlag, pp. 172-194, 1999.

[19] Y. Zhang, W. Lee, and Y.-A. Huang. Intrusion Detection Techniques for Mobile Wireless Networks, *ACM J. Wireless Net.*, vol. 9, no. 5, pp.545-556, 2003.

[20] Y. Zhang, W. Liu, W. Lou, and Y. Fang. Securing Mobile Ad Hoc Networks with Certificateless Public Keys. *IEEE Trans. Dependable Secur. Comput.*, vol. 3, no. 4, pp. 386-399, 2006.

[21] L. Zhou and Z.J. Haas. Securing Ad Hoc Networks, *IEEE Network Journal*, vol. 13, no. 6, pp. 24-30, 1999.