# Self-Organized Public Key Management for Mobile Ad Hoc Networks

Johann van der Merwe, Dawoud Dawoud and Stephen McDonald

***Abstract*— Mobile ad hoc networks (MANETs) offer communication over a shared wireless channel without any pre-existing infrastructure. Forming security associations in MANETs is more challenging than in conventional networks due to the lack of central authority. The main objective of this paper is to find a low complexity key management scheme that is suitable for self-organized MANETs. The proposed public key management scheme uses subordinate public keys and crypto-based identifiers to eliminate all forms of trusted third party. Nodes can create, disseminate and revocate their own keying material making the scheme self-organized and fully scalable. The paper proves the scheme to be secure in the Random Oracle and Generic model (ROM+GM).**

## I. Introduction

Mobile ad hoc networks (MANETs) eliminate the need for any fixed or pre-existing infrastructure by relying on the nodes to perform all network services. Pure MANETs are created solely by the end-users for a common purpose in an ad hoc fashion. Impromptu, self-organized MANETs can be visualized as a group of strangers, people who have never met before, coming together for a common purpose. These people have no prior relationship and share no common security information on their nodes. Users/nodes therefore have to establish security associations between them after network formation without the aid of *a priori* shared keying material or any form of common *off-line* trusted third party (TTP).

Several solutions for public key management schemes have already been proposed for MANETs [1] [2] [3] [4] [5]. From the existing solutions only [4] [5] succeed in eliminating the need for an off-line and on-line TTP.

The main objective of this paper is to propose a public key management scheme that is suitable for MANETs formed impromptu, with all forms of off-line and on-line TTPs eliminated. The proposed key management scheme, called Self-Organized Public Key Management (SelfOrgPKM) is based on a variant of the ElGamal type signature scheme, *subordinate public keys* and crypto-based identifiers [6] [7]. In the proposed scheme the nodes initialize themselves before joining the network. The scheme's operation is fully self-organized, with the burden of key management uniformly distributed between all network participants. Each node is thus its own authority domain, which is also our main assumption. The nodes establish security associations with their one-hop neighbors on the network layer during route establishment or on the application layer with a peer node on a need to know basis.

This work was supported by ARMSCOR, the Armaments Corporation of South Africa.

Johann van der Merwe, Dawoud Dawoud and Stephen McDonald are with the School of Electrical, Electronic and Computer Engineering, University of KwaZulu-Natal, South Africa (email: {vdmerwe, dawoudd, mcdonalds}@ukzn.ac.za).

*Subordinate public keys* are defined as public keys that are derived from a user's *self-generated* primary or base public/private key pair. We impose the following properties on subordinate public keys:

1) A valid subordinate public key can only be generated if the entity knows the base or primary private key.

2) The user can *self-generate* a renewed subordinate public key as frequently as needed.

3) The subordinate private key must be statistically independent of the base private key and other renewed subordinate private keys, i.e. compromise of a subordinate public key does not reveal any information about the user's base public key or any future renewed subordinate private keys.

4) There must exist a binding between the user's base public key and subordinate public key that supports non-repudiation.

The paper is organized as follows: In Section-II the related work is briefly surveyed. Section-III presents a variant on the generalized ElGamal type signatures as a strong cryptographic building block for subsequent schemes. Section-IV introduces a new subordinate public key generation scheme. In Section-V the new public key management scheme, SelfOrgPKM, for impromptu MANETs is proposed. Section-VI discusses the security and features of the proposed public key management scheme. Some conclusions are provided in Section-VII.

## II. Related Work

The majority of existing schemes, for example [1] [2] [3], are based on variations of a distributed certificate authority ($DCA$) that is held responsible for vouching for the authenticity of keying material. An off-line TTP is used to initialize the $DCA$ nodes. The collection of $DCA$ nodes, on the other hand, can be seen as a distributed on-line TTP. In contrast to conventional networks, the certificate authority has to be distributed to avoid a single point of attack [1].

Capkun *et al.* [4] present a self-organized public key management scheme based on Pretty Good Privacy (PGP) [8]. Similar to PGP, each node disseminates its own certificates and keeps a certificate repository comprising of the certificates of nodes in its local neighborhood. Users share their certificate repositories and mutually authenticate each other's certificate by finding a certificate chain linking their certificates.

Montenegro *et al.* [7] and Bobba *et al.* [9] use crypto-based-identifiers to bind node identifiers to public keys. The crypto-based addresses are used to protect the basic exchanges between nodes and to bootstrap the routing security mechanism,

effectively breaking the routing-security interdependency cycle and solving the address ownership problem.

Lately in [5], Capkun *et al.* have proposed a peer-to-peer key management scheme that relies on user mobility to bring nodes within each others transmission range which allows them to exchange their certificates without relying on a secure routing infrastructure. The fully self-organized version of the scheme requires nodes to use a secure side channel between the users' personal devices to authenticate each other and to setup shared session keys. The secret side channel can be a short range connectivity system such as infrared or a physical wire [5].

## III. Modified ElGamal Signature Scheme

In this section a *modified* ElGamal type signature scheme is presented, developed from the generalized ElGamal signature due to Horster *et al.* [10]. The presented ElGamal variant will be used as a strong cryptographic building block in subsequent proposed schemes.

### A. System parameter setup

The following system parameters are generated as usual:

| | |
|---|---|
| $p, q$ | two large primes, such that $q \mid (p-1)$. |
| $g$ | generator of the cyclic subgroup of order $q$ in $(Z)_p^*$. |
| $H(\cdot)$ | collision free one-way hash function. |
| $x_P$ | private key of user $P$. |
| $y_P$ | public key of user $P$, where $y_P = g^{x_P} \ mod \ p$. |

### B. Signature generation

User $P$ selects a random number $k \in [1, q-1]$ and computes a public commitment $r$ as: $r = g^k \ mod \ p$.

User $P$ signs an arbitrary message $m$ by solving the following congruence: $s \equiv x_P + [H(m \parallel r)]k \ mod \ q$

The set $(s, r)$ is the signature of user $P$ on message $m$.

### C. Signature verification

Any outsider can use user $P$'s public key $y_P$ to verify the validity of the signature $(s, r)$ for a message $m$ by checking if the following equation holds:

$$g^s = y_P r^{H(m \parallel r)} \ mod \ p \tag{1}$$

## IV. Proposed Subordinate Public Key Generation Scheme

The proposed subordinate public key generation scheme, based on the modified ElGamal signature variant presented in Section-III, borrows concepts from *parameter hidden* signature schemes [11].

The system parameters introduced in Section-III-A are applicable. It is assumed that party $A$ has generated its own public key/private key pair as follows: Party $A$ chooses random number $x_A \in_R [1, q-1]$ as its private key and computes its corresponding public key as $y_A = g^{x_A} \ mod \ p$.

Party $A$ can generate a subordinate public key from its base key pair $(x_A, y_A)$ that satisfies the properties defined in Section-I as follows:

- Party $A$ chooses random number $k_A \in_R [1, q-1]$ and computes $r_A = g^{k_A} \ mod \ p$.
- Party $A$ computes its new subordinate private key as $x'_A = x_A + H(KI_A)k_A \ mod \ q$, where the subordinate key information is defined as $KI_A = [ID_A \parallel y_A \parallel r_A \parallel SerNo \parallel IssueDate \parallel ValPeriod \parallel ExtInfo]$. Note that the contents of $KI_A$ can be altered based on the network policy, where $ID_A$ is the identity of party $A$, $SerNo$ a unique sequence number, $IssueDate$ the date of issueing the certificate, $ValPeriod$ the validity period and $ExtInfo$ some additional extension information.
- Finally party $A$ computes its corresponding subordinate public key as:

$$y'_A = g^{x'_A} = y_A (r_A)^{H(KI_A)} \ mod \ p \tag{2}$$

Party $A$ can renew its subordinate key pair with a self-organized subordinate key renewal procedure: Party $A$ simply chooses a new random number $k'_A \in_R [1, q-1]$ and computes its renewed subordinate private key as $x''_A = x_A + H(KI'_A)k'_A \ mod \ q$, where $KI'_A = [ID_A \parallel y_A \parallel r'_A \parallel SerNo + 1 \parallel IssueDate' \parallel ValPeriod' \parallel ExtInfo']$.

## V. Proposed Self-Organized Public Key Management Scheme (SelfOrgPKM)

The proposed public key management scheme for MANETs uses subordinate public keys (Section-IV) and crypto-based identifiers [6] [7] as strong cryptographic building blocks to setup security associations between nodes with minimal communication and computational overhead. The bootstrapping of the security service does not require any form of off-line or on-line TTP, which is consistent with the characteristics of self-organized impromptu MANETs.

The operation of SelfOrgPKM is divided into a node initialization phase which is executed by each node before the node joins the network and a post-initialization which executes during network operation.

### A. Initialization Phase of SelfOrgPKM

Each node $P_i$, for $(1 \leq i \leq n)$ creates a base public/private key pair $(x_i, y_i)$ by choosing a random number $x_i \in_R [1, q-1]$ as its base private key and computes its corresponding public key as $y_i = g^{x_i} \ mod \ p$. It is assumed that each node has an authentic image of the system parameters, as specified in Section-III-A.

Each node generates a unique identifier ($ID_i$) that is bound to its base public key $y_i$ as follows:

$$ID_i = H(y_i) \tag{3}$$

SelfOrgPKM requires $ID_i$ to be used as the nodes network address or as a fixed part of the address. Note that this requirement places no constraint on the structure of the network addresses: The entire hash output, $ID_i$, can be used in MANETs with flat static addresses or only a part of the output can be used in MANETs with dynamic addressing.

Each node $P_i$ uses its base public/private key pair $(x_i, y_i)$ to generate a subordinate public/private key pair $(x'_i, y'_i)$ as specified in Section-IV.

Note that $P_i$'s base key pair $(x_i, y_i)$ is never used for real communication. Rather, each $P_i$ uses its subordinate key pair $(x'_i, y'_i)$ for securing actual communication.

To obtain an explicitly authentic key pair each node uses its newly obtained subordinate private key $x'_A$ to sign the key information contents $KI_i$ (concatenated with its subordinate public key $y'_i$) via the modified ElGamal signature scheme presented in Section-III. This is equivalent to the self-certificates proposed by Lee *et al.* [12], which are used to explicitly authenticate self-certified public keys. Node $P_i$'s self-certificate can then be define as: $SelfCert'_i = [KI'_i \parallel y'_i \parallel \alpha'_i \parallel \beta'_i]$, where $(\alpha'_i, \beta'_i)$ is the appended signature on $KI'_i \parallel y'_i$.

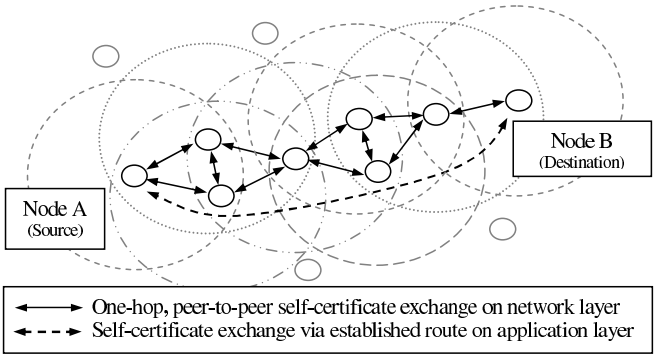


Fig. 1. SelfOrgPKM certificate exchange

### B. Post Initialization Phase of SelfOrgPKM

The post initialization phase commences after network formation. Each node must perform the initialization phase, presented in Section-V-A, before joining the network.

*1) Certificate exchange and authentication:* Certificate exchange takes place between nodes on a peer-to-peer, need to know basis. Nodes setup a bidirectional security association by interchanging their renewed self-certificates, $SelfCert'$. SelfOrgPKM requires all nodes to exchange self-certificates with their one-hop neighbors on the network layer: nodes within each others transmission range exchange their certificates during route establishment. We will limit our scope to on-demand routing without bounding our scheme to a specific routing protocol. The source node starts with a broadcast route request as usual with its renewed self-certificate $SelfCert'$ appended. Two unicast messages are needed for subsequent certificate exchange if the source and neighboring node have not done so already: one message from the neighboring node and one message from the source node. This process continues until the route request reaches the destination node, hence the neighboring nodes will remove the source node's certificate, append their own certificate and broadcast the route request to their neighboring nodes.

It is trivial to see that the one-hop network layer certificate exchange mechanism makes the security scheme independent of the routing-security interdependence cycle as defined in [9].

The following example, defined in Figure-1, explains certificate exchanges at the application layer: assume $Node_A$ wants to communicate securely with $Node_B$. In the first round $Node_A$ sends to $Node_B$ a $CertRequest$ requesting $SelfCert'_B$ from $Node_B$ over the established route. Note that $CertRequest$ contains the certificate $SelfCert'_A$ of $Node_A$. If $Node_B$ grants the request it replies in the second round with $SelfCert'_B$. Note that this two round procedure requires no synchrony between $Node_A$ and $Node_B$. The self-certificates and subordinate public keys of $Node_A$ and $Node_B$ are authenticated as follows:

1) Each node implicitly authenticates the base public key of its peer node by checking if Equation-3 holds.

2) Next, the peer nodes implicitly authenticate the subordinate public key of their peers by checking if Equation-2 holds.

3) Finally each node validates the self-certificate of its peer node, $SelfCert'_i$, by verifying the signature $(\alpha'_i, \beta'_i)$ on $[KI_i \parallel y'_i]$. This explicitly authenticates both the base public key $y_i$ of $Node_i$ and the subordinate public key $y'_i$. If the node identifier matched the hash output the node is assured that the node identifier/address has not been spoofed.

If all three of the above verifications hold then the subordinate public key $y'_i$ is explicitly authentic and securely bound to the base public key $y_i$ which in turn is securely bound to the nodes *statistically* unique identifier/address.

If the certificate transfer fails due to error-prone connectivity or unsuccessful authentication, $Node_A$ will retry by resending $CertRequest$.

For efficiency reasons nodes can use symmetric key schemes to secure all subsequent messages. This can easily be achieved by using the available authenticated public keys to establish a session key between peers.

*2) Certificate revocation:* SelfOrgPKM makes use of a self-revocation system based on a self-organized subordinate key renewal procedure. As mentioned in Section-V-A, nodes do not use their base key pair for any real communication, but must derive a subordinate key pair $(x'_i, y'_i)$ from the base key pair which is then used for actual communication. This significantly reduces the chance of a successful attack on a node's base key pair [12]. The self-organized key renewal process given in Section-IV can be used by the node to obtain a renewed key pair $(x''_i, y''_i)$ at any point in time during the post initialization operation of the network. The node will thus derive a new private key $x''_i = x_i + H(KI'_i)k'_i \ mod \ q$ and generate a new self-certificate $SelfCert''_i == [KI'_i \parallel y''_i \parallel \alpha''_i \parallel \beta''_i]$. The renewed certificate $SelfCert''_i$ can be sent to the node's frequent contacts or offered to other nodes on communication initialization. Since nodes are responsible for their own keying material they can renew their subordinate key pair as frequently as desired. Nodes will however most likely renew their key pair in two instances: when they suspect that their subordinate private key as been compromised or when their set validity periods $ValPeriod'$ have expired.

## VI. Discussion on the Security and Features of SelfOrgPKM

The proposed public key management scheme for MANETs, presented in Section-V, makes use of subordinate public keys and crypto-based identifiers as building blocks to

effectively eliminate the need for any form of off-line or on-line TTP. The availability of an off-line TTP is fundamentally against the characteristics of impromptu MANETs. This makes schemes such as [1] [2] [3] unsuitable for impromptu MANETs. The weaknesses of these existing schemes extends into network formation. They use a distributed certificate authority ($DCA$) as an on-line TTP which can be attacked. Our proposed scheme avoids these weaknesses by using a fully distributed system where each node becomes its own authority domain.

The existing schemes that take the characteristics of impromptu MANETs into consideration have the following main weaknesses:

1) The PGP approach presented in [4] only provides weak certificate authentication and may fail to provide certificate chains between all node pairs in the network.

2) The major weakness of the crypto-based identifier approach [7] [9] is that users cannot revocate their public keys without changing their network addresses and/or identifiers [5].

3) The peer-to-peer key management scheme in [5] has a significant time delay in the setup of the security associations.

Our proposal inherits the benefits of crypto-based identifiers [7] [9] as a means of solving the address ownership problem. Subordinate public keys introduced in Section-IV are used for real communication leaving an adversary with a brute-force attack as the only option to compromise a node's base public/private key pair and/or identifier. The subordinate key pairs can also easily be renewed without having to modify the base public key pair which keeps the nodes' identifiers constant.

### A. On the security of SelfOrgPKM

The proposed public key management scheme introduced in Section-V use as secure building block the modified generalized ElGamal type signature variant presented in Section-III. It is noted that the modified ElGamal type signature variant is essentially the Generalized ElGamal Scheme: $GES = (M.EG\text{II}.3.\sigma(1), r, s, 1, h(m, r))$ [10] and therefore benefits from the same security properties. Compared to the digital signature standard (DSS) [13], the proposed modified variant has equivalent security, but outperforms DSS and most of the other ElGamal variants in terms of computational efficiency. The security analysis on the ElGamal signature variants presented in [10] and the majority of variants proposed in literature are in fact heuristic, i.e. the security analysis considers known attacks and informally argues that the ElGamal variant under investigation is resistant to these attacks.

In the next subsection the security of our public key management scheme is proved in a widely accepted cryptographic model.

*1) Security proof of proposed scheme:* In the following proof we refer to the combined security model, the Random Oracle and Generic Model (ROM+GM), proposed by Schnorr *et al.* [14] [15].

In the first part of the security proof for the proposed public key management scheme, SelfOrgPKM, it will be shown that the modified generalized ElGamal type signature variant presented in Section-III, is secure against the *one-more signature forgery* attack [14] in the ROM+GM model.

*Theorem 1:* Let a generic adversary $\mathcal{A}$ interact with a signer and be given $g$, the public key $y$ and an oracle for $H$. $\mathcal{A}$ performs $t$ generic steps which include $l$ sequential signer interactions. With a probability space consisting of $y$, $H$ and coin flips of the signer, it is not possible for $\mathcal{A}$ to produce $l+1$ signatures with a probability better than $\frac{\binom{t}{2}}{q}$.

In the following proof *Lemma 1* and *Lemma 2* are those defined and proved in [14].

*Proof:* [following Schnorr *et al.* [14]]

As given by *Lemma A* defined below, the group element $f_{i'} = g^{\frac{s'_i}{c'_i}} g^{-\frac{x}{c'_i}} = g^{\langle \alpha_{\mathbf{i}'}, (1,x,\mathbf{k}) \rangle}$ for an arbitrary $i \leq t'$. $\mathcal{A}$ receives hash query $c'_i = H(m \parallel g^{\frac{s'_i}{c'_i}} g^{-\frac{x}{c'_i}})$ and needs to find $s'_i$ which satisfies Equation-6. The adversary $\mathcal{A}$ is thus required to solve a linear polynomial $x + c'_i \langle \alpha_{\mathbf{i}'}, (1, x, \mathbf{k}) \rangle$ at $(x, \mathbf{k})$. By *Lemma 2* due to [14], $x$ is statistically independent from $(\alpha_{\mathbf{i}'}, (1, x, \mathbf{k}))$, excluding prior collisions $f_j = f_k$. By *Lemma 1* presented in [14], it is known that such collisions will only occur with an upper bound probability of $\frac{\binom{t'}{2}}{q}$. On the other hand, by *Lemma A*, adversary $\mathcal{A}$ must choose $c_1, \ldots, c_l$ for each signature $(m'_i, c'_i, s'_i)$ that satisfies Equation-4 such that $x$ cancels out. In the case of a sequential attack, without any collisions among the computed group elements $f_1, \ldots, f_{t'}$, the system of $l+1$ equations for $c_1, \ldots, c_l$ is solvable with an upper bound probability of $\frac{\binom{t''}{2}}{q}$, where $t''$ denotes the number of queries to $H$ [14]. It follows from $\frac{\binom{t'}{2}}{q} + \frac{\binom{t''}{2}}{q} \leq \frac{\binom{t}{2}}{q}$, that $\frac{\binom{t}{2}}{q}$ is the highest probability for $\mathcal{A}$ to succeed in a sequential, *one-more signature* attack on the signature scheme presented in Section-III. □

*Lemma A:* Let the triplet $(m'_i, c'_i, s'_i)$ be a signature with a probability better than $\frac{1}{q}$. The $c'_i$-coordinate then coincides with the value $H(m \parallel f)$ corresponding to the hash query $(m \parallel f)$. From Equation-1, $g^k = g^{\frac{s}{c}} g^{-\frac{x}{c}}$. The hash query $(m \parallel f) \in G \times M$, satisfies $c'_i = H(m \parallel f) = H(m \parallel g^{\frac{s'_i}{c'_i}} g^{-\frac{x}{c'_i}})$, where the group element $f = f'_i$ for some arbitrary $1 \leq i' \leq t'$. The parameters $(m'_i, c'_i, s'_i)$ also satisfy:

$$c'_i = \frac{1}{-\alpha_{i',1} + \sum_{k=1}^{l} \left[\alpha_{i',k}\, c_k^{-1}\right]} \tag{4}$$

$$s'_i = c'_i \left[\alpha_{i',0} + \sum_{k=1}^{l} \alpha_{i',k} \frac{s_k}{c_k}\right] \tag{5}$$

In the following proof *Lemma 2* is defined and proved in [14].

*Proof:* [following Schnorr *et al.* [14]]

Since $1 \leq i' \leq t'$ denotes the index of $f$ among the computed group elements $f_1, \ldots, f'_t$, the group element can be written as $f_{i'} = g^{\frac{s'_i}{c'_i}} g^{-\frac{x}{c'_i}} = g^{\langle \alpha_{\mathbf{i}'}, (1,x,\mathbf{k}) \rangle}$. It follows from

the previous equation and $k_k = \frac{s_k}{c_k} - \frac{x}{c_k}$, that:

$$s'_i = x + c'_i \log_g \left[ g^{\frac{s'_i}{c'_i}} g^{-\frac{x}{c'_i}} \right] = x + c'_i \langle \alpha_{i'}, (1, x, \mathbf{k}) \rangle \quad (6)$$

$$\begin{aligned} s'_i &= c'_i \left[ \alpha_{i',0} + \sum_{k=1}^{l} \alpha_{i',k} \frac{s_k}{c_k} \right] + \\ & \quad x \left[ 1 + c'_i \left[ \alpha_{i',1} - \sum_{k=1}^{l} \alpha_{i',k} \frac{1}{c_k} \right] \right] \end{aligned} \quad (7)$$

In order for the generic adversary $\mathcal{A}$ to calculate the correct $s'_i$, $\mathcal{A}$ must find $c'_i$ such that $x$ cancels out. $\mathcal{A}$ must therefore select $c_1, \ldots, c_l$ that satisfies Equation-4.

If $x$ cancels out, $s'_i$ can be computed by $\mathcal{A}$ as specified by Equation-5.

In the case that $x$ does not cancel out in the equality given by Equation-7, the equality will only hold with probability $\frac{1}{q}$ since $x$ is statistically independent from non-group data by *Lemma 2* presented in [14].

□

It is trivial to see that Equation-3 proves the base public key $y_i$, of an arbitrary network participant $P_i$ *implicitly* authentic within ROM+GM. The second part of the security proof will thus focus on the *explicit* authenticity of the base public key $y_i$ and subordinate public key $y'_i$, generated via the proposed subordinate key generation scheme (Section-IV). It is shown that the security of the subordinate public key and the explicit authenticity of both the base and subordinate public keys, are directly dependent on the security of the signature scheme on which the self-certificate generation procedure (Section-V-A) is constructed. Note that the self-certificate generation procedure provides a zero knowledge proof of knowledge of $x'_i$ making the subordinate public key pair explicitly authentic. We also claim that it serves as a key confirmation procedure for the base private key $x_i$.

*Theorem 2:* The proposed subordinate key generation scheme, integrated with the self-certificate generation procedure is as secure as the signature scheme on which it is based in the ROM+GM security model. A signature with the subordinate private key $x'_i$ also proves both the base private key $x_i$ and the subordinate private key $x'_i$ explicitly authentic.

*Proof:* [Informal Survey]

From any entity's perspective Equation-2 can only provide *implicit* authentication of $y'_i$, i.e. the verification procedure gives no assurance that $P_i$ knows the corresponding private key $x'_i$. The authenticity of the subordinate public key only becomes known when $P_i$ uses it for a cryptographic procedure which inherently provides a zero-knowledge (ZK) proof of knowledge of $x'_i$.

An adversary $\mathcal{A}$ that wants to produce a forged subordinate public key must compute a public key $y'_{\mathcal{A}}$ that satisfies:

$$y'_{\mathcal{A}} = y_i \cdot (r_{\mathcal{A}})^{H(KI_{\mathcal{A}})} \; mod \; p \quad (8)$$

$\mathcal{A}$ does not know $\log_g y'_{\mathcal{A}}$ and will consequently fail to produce a valid ZK proof. This serves as motivation for introducing self-certificate generation in Section-V-A, which effectively serves as a ZK proof. It will thus be appropriate to access the security of the proposed subordinate public key generation protocol in conjunction with the signature $(\alpha'_i, \beta'_i)$ on $m_i = [KI_i \parallel y'_i]$ as described in Section-V-A. It is noted that $(\alpha'_i, \beta'_i)$ is produced via the proposed signature scheme presented in Section-III. The verification equation on $(\alpha'_i, \beta'_i)$ is given as:

$$g^{\alpha'_i} = y'_i \cdot (\beta'_i)^{H(m_i \| \beta'_i)} \; mod \; p, \quad (9)$$

Substituting Equation-2 into Equation-9 yields:

$$g^{\alpha'_i} = y_i \cdot (r_i)^{H(KI_i)} \cdot (\beta'_i)^{H(m_i \| \beta'_i)} \; mod \; p, \quad (10)$$

which has the following signature equation:

$$\begin{aligned} \alpha'_i &= x_i + H(KI_i)(k_i) \\ &\quad + H(m_i \parallel \beta'_i)(\log_g \beta'_i) \; mod \; q \end{aligned} \quad (11)$$

An entity which explicitly authenticates $y'_A$ via Equation-2 and Equation-9, indirectly verifies Equation-11 in two steps. From Equation-10 and Equation-11 it is clear that an adversary $\mathcal{A}$ can only generate a forged subordinate public key with an upper bound probability of $\frac{\binom{t}{2}}{q}$ in the ROM+GM security model (by *Theorem 1*). Furthermore it shows that the verifier of Equation-2 and Equation-9 can be assured that the party with $ID_i$, generated via Equation-3, knows the base private key $x_i$ corresponding to $y_i$. □

Finally we prove the proposed peer-peer key management scheme secure as a whole.

*Theorem 3:* The proposed public key management scheme, SelfOrgPKM, is as secure as the signature scheme on which it is based in the ROM+GM security model.)

*Proof:* [Informal Survey]

As SelfOrgPKM is based on an independent combination of the subordinate public key generation scheme (Section-IV) and crypto-based identifiers [6] [7]; the proof for *Theorem 3* follows from *Theorem 2* and the properties of the ideal hash oracle in ROM+GM. □

### B. On the efficiency of SelfOrgPKM

SelfOrgPKM is fully distributed, preserving the symmetric relationship between nodes as required in MANETs.

*1) Efficiency of SelfOrgPKM initialization phase:* The initialization phase is performed by each node before joining the network and therefore has no impact on network performance. This process should however still be as efficient as possible. Each $P_i$ performs 4 exponentiations ($exp$), 3 random number generations ($R_{gen}$) and 3 hash computations ($H(\cdot)$) (The 2 multiplications and 2 summations have insignificant impact on the time complexity in comparison with the exponentiations). The initialization phase has no communication cost.

*2) Efficiency of SelfOrgPKM post initialization phase:* The on-line post initialization phase of SelfOrgPKM results in little overhead for each node. A node renewing its self-certificate has to perform only two signature generations and two exponentiation to compute its renewed subordinate public key with a total cost of (3 $exp$, 2 $R_{gen}$ , 2 $H(\cdot)$). Any node can verify another nodes' self-certificate with a computational cost of (3 $exp$, 1 $H(\cdot)$) and only (3 $exp$) for all subsequent

verifications since the base public key has to be certified only once.

Self-certificate exchanges on a peer-to-peer basis (on the application and network layers) are the only communication overhead imposed on the network by the proposed scheme. A certificate exchange procedure on the application layer only takes two asynchronous rounds with one unicast message each. One extra broadcast round is needed on the network layer since the node making the route request does not have information referring to its neighbours.

### C. Evaluation of SelfOrgPKM

The effectiveness of the proposed public key management scheme was investigated through simulations in the ns-2 simulator (release 2.28) [16] using the OpenSSL cryptographic library (version 0.9.7e) [17] to implement the basic modular arithmetic and system parameter generation. The implementation supported the mathematical correctness of the cryptographic design and showed that nodes are guaranteed of exchanging their certificates except if nodes become permanently disconnected.

## VII. Conclusion

The paper proposes a novel public key management scheme for impromptu mobile ad hoc networks, called Self-Organized Public Key Management (SelfOrgPKM). The scheme has low implementation complexity and provides self-organized mechanisms for certificate dissemination and revocation without the needs for any form of off-line or on-line authority.

The fully distributed scheme is superior in communication and computational overhead with respect to its counterparts. All nodes send and receive the same number of messages and do the same number of computations. SelfOrgPKM therefore preserves the symmetric relationship between the nodes. Each node is its own authority domain which provides an adversary with no convenient point of attack.

SelfOrgPKM solves the classical routing-security interdependency and address ownership problem by providing a strong one-to-one binding between a user's certificate information and public key.

The SelfOrgPKM is furthermore proven secure in the ROM+GM model (*Theorem 3*), which is to the best of the authors knowledge the first public key management scheme for self-organized impromptu MANETs with such a strong notion of security.

The paper also introduces two generic cryptographic building blocks as the basis of SelfOrgPKM: 1) A variant on the ElGamal type signature scheme developed from the generalized ElGamal signature scheme due to Horster *et al*. The modified scheme is one of the most efficient ElGamal variants, outperforming most of the other variants; and 2) A subordinate key generation scheme.

The paper introduces the novel notion of *subordinate public keys*, which allow the users of SelfOrgPKM to performed self-organized self-certificate revocation without changing their network identifiers/addresses. The presented ElGamal variant and subordinate key generation scheme were also proved to be secure in ROM+GM (*Theorem 1* and *Theorem 2*) without making any unrealistic assumptions.

The only operation of SelfOrgPKM affecting the network is the pairwise exchange of certificates. The cryptographic correctness, low implementation complexity and effectiveness of SelfOrgPKM was verified though simulation using ns2 and OpenSSL.

## References


[1] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," *IEEE Network: special issue on network security*, vol. 13, no. 6, pp. 24–30, 1999.

[2] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing Ad Hoc Wireless Networks," in *proc. Seventh International Symposium on Computers and Communications (ISCC'02)*, July 1-4 2002.

[3] S. Yi and R. Kravets, "MOCA: Mobile certificate authority for wireless ad hoc networks," in *proc. of the 2nd Annual PKI Research Workshop (PKI 2003)*, April, 28-29 2003.

[4] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-Organized Public-Key Management for Mobile Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 2, no. 1, pp. 52–64, 2003.

[5] S. Capkun, J. Hubaux, and L. Buttyan, "Mobility Helps Peer-to-Peer Security," *IEEE Transactions on Mobile Computing*, 2004, to appear.

[6] G. O'Shea and M. Roe, "Child-proof authentication for MIPv6 (CAM)," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 2, pp. 4–8, 2001.

[7] G. Montenegro and C. Castelluccia, "Crypto-based Identifiers (CBIDs): Concepts and Applications," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 97–127, 2004.

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

[9] R. B. Bobba, L. Eschenauer, V. D. Gligor, and W. Arbaugh, "Bootstrapping Security Associations for Routing in Mobile Ad-Hoc Networks," in *proc. IEEE Global Telecommunications Conference c*, December 2003.

[10] P. Horster, M. Michels, and H. Petersen, "Generalized ElGamal signatures for one message block," in *proc. 2nd Int. Workshop on IT-Security*, September, 22-23 1994.

[11] ——, "Meta-Message Recovery and Meta-Blind Signature Schemes Based on the Discrete Logarithm Problem and their Applications," in *proc. Advances in Cryptology - ASIACRYPT'94*, 1994.

[12] B. Lee and K. Kim, "Self-certificate: PKI using Self-certified Key," in *proc. Conf. on Information Security and Cryptology (CISC'00)*, November, 25 2000.

[13] National Institute of Standards and Technology (NIST), "Digitial Signature Standard (DSS)," U.S. Department of Commence, Federal Information Processing Standards Publication FIPS PUB 186, May, 19 1994.

[14] C. P. Schnorr and M. Jakobsson, "Security of Discrete Log Cryptosystems in the Random Oracle and Generic Model," in *proc. The Mathematics of Public-Key Cryptography*, June, 12- 17 1999.

[15] ——, "Security of Signed ElGamal Encryption," in *proc. Advances in Cryptology - ASIACRYPT '00*, December,3-7 2000.

[16] "The Network Simulator - NS-2," available at www.isi.edu/nsnam/ns.

[17] "OpenSSL cryptography library," available at www.openssl.org.



**Johann van der Merwe** received his BSc degree in Electronic Engineering from the University of Natal in 2003. In 2004, he joined the communications research group in the School of Electrical, Electronic and Computer Engineering at the University of KwaZulu-Natal, where he is working towards his MSc degree. His current research interests include security issues in ad hoc networks, with particular focus on key management, threshold cryptography, secret sharing and digital signature schemes.