Resisting to False Identities Attacks to the Public-Key Management System for Wireless Ad Hoc Networks

Eduardo da Silva, Renan Fischer e Silva, and Luiz Carlos P. Albini

NR2, Informatics Depto., Federal University of Paraná, Curitiba, Brazil
eduardos@inf.ufpr.br, renan@inf.ufpr.br, albini@inf.ufpr.br

Abstract. Cryptography is widely known as the best technique to provide security on data communications in all kinds of networks. Cryptographic methods rely on keys to perform their operations, such as encryption, decryption, and signature. In Wireless Ad Hoc Networks (WANETs), key management is a critical service as it must handle all security threats in a self-organized and decentralized way. Several kinds of attacks can compromise the key management on WANETs, such as Sybil and bad mouthing. This article presents the enhanced VKM, called *e*-VKM, a virtualization-based key management system resistant to Sybil and bad mouthing attacks. *e*-VKM is proposed to work on scenarios in which nodes can be preloaded with secure information before joining the system. Examples of these scenarios include but are not limited to sensor networks, meeting conferences, battlefield operations or health care solutions. Results show that *e*-VKM is highly resistant to Sybil attacks and bad mouthing, presenting 100% of resistance even under 20% of attackers.

Keywords: Wireless Ad Hoc Networks, Key Management, Security, Virtualization.

1 Introduction

Wireless Ad Hoc Networks (WANETs) are composed by a set of mobile devices (nodes) which communicate via a shared wireless medium. Nodes of these networks may be either mobile or stationary, and do not rely on any fixed infrastructure or centralized control [1]. WANETs support a wide range of applications ranging from military to civilian ones. Application examples include rescue or military operations, disaster scenarios, ubiquitous health care, sensor networks and conference meetings. However, due to their characteristics, WANETs are highly vulnerable to passive and active attacks [2].

Cryptography is widely known as the main technique to secure data communications [3]. It provides information integrity, authenticity, non-repudiation and confidentiality. Cryptographic techniques depend on keys, which are information related to the parties and are used within cryptographic algorithms on encryption and decryption operations, and on digital signatures. The secure administration of such keys, called "key management", must be safe against threats which try to compromise the confidentiality and authenticity, and must avoid their non-authorized use [3]. Key management in WANETs must consider the dynamic topology and be self-organized and decentralized [4]. Further, it must: not have a single point of failure; be compromise-tolerant;

K. Pentikousis et al. (Eds.): MONAMI 2011, LNICST 97, pp. 344–357, 2012.

be able to revoke keys from compromised nodes; update keys from non-compromised nodes; be efficient in terms of storage, computation and communication.

Many key management systems have been proposed for WANETs [4]. It is possible to classify them in identity-based [5], chaining-based [6–8], cluster-based [9, 10], mobility-based [11] and virtualization-based [12, 13]. The foremost scheme so far is the Self-Organized Public Key Management System, called *PGP-Like* in this work [6, 8]. It is fully distributed and self-organized while following the concepts of PGP [14], in which each node creates its own keys and issues certificates to other ones it trusts. Nodes must periodically exchange certificates with neighbors and keep them in local repositories.

The PGP-Like characteristics make it very attractive to WANETs, though they also make it highly vulnerable to Sybil attacks [15, 16]. If an attacker which maintains a correct behavior during a while, creates a false identity and issues a certificate to it, all nodes that trust in the attacker will also trust in the false identity. Note that the only security mechanism that PGP-Like has is a certificate conflict detection one, in which a node only classifies a certificate as conflicting, non-conflicting, or false.

Even though protocols for WANETs should be self-organized and fully distributed, several application scenarios allow nodes to be pre-configured before joining the network. For example: sensors may be configured before scattered into the environment; soldier equipments can be charged with confidential information before he advances in a battlefield; meeting attendants might receive all required information during the registration; etc. In these scenarios it is possible to consider the existence of an external entity preloading nodes before either network formation or joining the network.

In such a context, in [12], a preliminary version of the Virtual Key Management (VKM) was presented. VKM is a virtualization-based scheme; the virtualization layer, or virtual structure, represents the trust relations between nodes. Based on the virtual structure, nodes issue mutual certificates. Similar to the PGP-Like, nodes perform key authentications through certificate chains. Even though its characteristics make it more resistant to Sybil attacks than PGP-Like, results are not satisfactory. In fact, almost 100% of the authentications in VKM contain false identities with 40% of attackers in the system. Further, [12] does not mention update or revoke operations of the VKM.

This work introduces the enhanced VKM (*e*-VKM). *e*-VKM is the first key management for WANETs based in a traceable and verifiable multi-signature scheme, such as [17]. All certificates are issued using the multi-signature scheme to provide a very high resistance against Sybil [18] and bad mouthing [19] attacks. *e*-VKM uses the same virtualization layer as VKM, called "virtual structure", to indicate the trust between nodes. *e*-VKM is the first key management scheme for WANETs completely resistant to the Sybil and Bad Mouthing attacks. In fact, it maintains its performance and effectiveness in scenarios with up to 40% of attackers in the network without considerably increasing the overhead. *e*-VKM was also compared with PGP-Like to illustrate its efficacy and potentiality.

The rest of this paper is organized as follows: Section 2 describes the main attacks which may compromise key management schemes in WANETs; Section 3 introduces the Self-Organized Public Key Management System; Section 4 describes the *e*-VKM

scheme; Section 5 presents the evaluation of *e*-VKM and a comparison with PGP-Like and the original VKM; finally, Section 6 contains the conclusions and future work.

2 Attacks on WANETs

WANETs are susceptible to many security issues as a consequence of their natural characteristics and properties. Multihop communication, lack of infrastructure, limited resources and dynamic topology make them vulnerable to several kinds of passive and active attacks, in which attackers can eavesdrop or delete packets, modify packet contents, forge messages, or even impersonate other nodes [2,20]. Table 1 [21] summarizes the main attacks for WANETs classified according to the network protocol stack.

Table 1. Types of active attacks on WANETs

Layer	Attack	Description
Physical	Exhaustion	Repeated retransmission in order to exhaust resources of other nodes
	Jamming	Transmissions on the radio frequency to deny the use of the channel
Link	Collision	Attacker generates collisions to deny link usage
Network	Wormhole	Two attackers deviate packets through a side-channel
	Blackhole	Attacker discards received routing messages
	Selective forward	Packets which must be forwarded are selectively discarded
	Sinkhole	Traffic from the network are deviated to pass through an adversary
Transport	SYN Flooding	Several connection requests to a target, overwhelming its resources
Multi-layers	Bad Mouthing	Attacker performs a false recommendation against another node
	Sybil	Attacker uses multiple fake identities
	Lack of Cooperation	Node which does not cooperate in network activities

Among these attacks, the impersonation and bad mouthing ones can be very harmful to a key management system. Even though key management systems are developed to protect the network against attacks, they do not consider attacks against themselves. Without loss of generality, this work focuses on these attacks.

Sybil attacks consist in malicious nodes using false identities in a unique device [18]. These false identities can be fabricated or spoofed from honest nodes. A Sybil attacker can use false identities in order to manipulate keys and certificates, deceiving systems which use the key management service. Further, in self-organized and distributed schemes, Sybil attackers might control the key management scheme, allowing other untrusted nodes to join the system [22, 23]. Thus, Sybil nodes can violate confidentiality, authentication and non-repudiation security principles.

Bad mouthing attacks consist of malicious nodes providing false accusations to defame honest nodes and revoke their keys [19]. In a key management scheme which considers information and recommendation of nodes to update or revoke the keys of other nodes, this kind of attack can be highly harmful. A large amount of work can be found in the literature to deal with these threats, for example: techniques to detect Sybil attacks can be found in [18, 24]. However, these studies are focused on routing protocols.

3 Self-organized Public Key Management System

The Self-Organized Public Key Management System, called in this work PGP-Like, is a public key management scheme which uses certificate chains [6,8]. Private and public keys of nodes are created by the nodes themselves following the PGP concepts [14]. In addition, each node issues public key certificates to other ones it trusts. In PGP-Like, if a node u believes that a given public key K_v belongs to a given node v, it issues a certificate binding K_v to the node v, $(v,K_v)_{prK_u}$, in which prK_u is the private key of node u. This certificate is stored in the local certificate repositories of both nodes, u and v. Moreover, each node periodically exchanges its repository with its physical neighbors.

Public keys and certificates are represented by a directed graph. A directed edge between two vertexes K_u and K_v, $(K_u \rightarrow K_v)$ denotes a certificate, signed by node u, binding K_v to node v. A path connecting two vertexes K_u and K_v is represented by $(K_u \rightsquigarrow K_v)$. Each node u maintains an updated local certificate repository, G_u, and a non-updated local certificate repository, G_u^N [6]. The non-update local certificate repositories contain the certificates which have expired.

When node u wants to authenticate the public key K_v of node v, it firstly tries to find a path from K_u to K_v in G_u. If $\exists(K_u \rightsquigarrow K_v) \in G_u$, node u authenticates it. If $\neg\exists(K_u \rightsquigarrow K_v) \in G_u$, node u creates $G' = G_u \cup G_v$, and it tries to find $(K_u \rightsquigarrow K_v) \in G'$. If such a path exists, the authentication succeeds.

The path found is a certificate chain. Certificate chains represent the trustworthiness between nodes and are called trust chains. Note that trust chains are weak authentications, as they assume that trust is transitive. Unfortunately, ensuring a valid transitive trust with more than two nodes in the chain is very difficult [25]. Thus, if any node in the chain is compromised, all other nodes of this chain might obtain false authentications.

The use of certificate chains makes PGP-Like highly vulnerable to Sybil attacks, as shown in [15]. An attacker, node x, can create a false identity m and issue a certificate binding k_m to m. All nodes that trust x will also trust m. Thus, if node x maintains a correct behavior during considerable time, several units will probably trust it and the false identity will be spread over due to the certificate exchange mechanism.

4 Enhanced Virtual Key Management System

This section presents the enhanced Virtual Key Management System (e-VKM). e-VKM uses the same virtualization layer as VKM, called "virtual structure", to indicate the trust between nodes. However, the remaining of the e-VKM is completely different, as it is the first key management for WANETs based in a traceable and verifiable multi-signature scheme. All certificates are issued using the multi-signature scheme. Table 2 summarizes the notation used. Section 4.1 presents an overview of e-VKM, section 4.2 discusses the multi-signature concepts and section 4.3 details the e-VKM operation.

4.1 System Overview

e-VKM is focused on wireless ad hoc network, stationary or mobile, consisting of a set of n nodes. Nodes have similar functionalities, and must contribute with the network maintenance and operations, including the key management. Two nodes (i, j)

Table 2. Used notation

Notation	Description
i	node identity
N	PKI nodes set
pk_i	public key of a given node i
sk_i	private key of a given node i
$VN(i)$	set of virtual neighbors of node i
$C^i_{VN(i)}$	public key certificate of node i generated by its virtual neighbors
T_v	the expiration time of a certificate
$SIGN[a]_{Sw}$	signing some given information a with S_w
$AUTH[X_i \rightsquigarrow X_v]$	X_i is authenticating p_v of X_v
$a\|\|b$	some given information a is concatenated with some given information b
G_i	repository of updated certificates of X_i
G^N_i	repository of non-updated certificates of X_i
$\|Z\|$	Size of a given set Z

are considered physical neighbors if they are inside each other's communication range. Further, without loss of generality, the system assumes that all nodes have the same communication range. Finally, due to the self-organized and autonomous characteristics of WANETs, no node has complete knowledge of the physical network topology.

In addition, *e*-VKM focuses on application scenarios which allow an external entity to preload nodes with basic parameters, such as the virtual structure formation rule, before network formation. In fact, such a requirement is suitable for many scenarios, in which nodes can be registered before actually using the network resources. For example, in a conference meeting users must register themselves to take part in the conference. In this moment, nodes might be preloaded with the required information. Other examples include: a military network in which users might be preloaded before going into a battlefield, and a sensor network in which sensors might be preloaded by their users before being scattered. These are just some examples showing that the preload requirement can be found in some real WANET scenarios. Note that, the external entity must be present just when a node wants to join the system. After that, the assistance of the external entity is not necessary, and nodes are able to perform all operations by themselves, including the key management.

e-VKM uses a virtualization layer, called virtual structure, to indicate the certificate chains formation. The virtual structure is represented by a directed graph $L = (N, E)$, which is unrelated to the physical network topology. Set N represents the n nodes and set E represents the virtual links. A virtual link $(i, j) \in E$ indicates that node i should obtain and keep the public key pk_j of node j. Two nodes connected by a virtual link are called virtual neighbors.

It should be noticed that *e*-VKM is, to some extent, independent of the graph implementing the virtual structure, i.e. it can use any connected graph as the virtual structure. However, it appears that the graph should be regular (i.e. the degree must be the same for all nodes) to ensure that the number edges is the same for all nodes. The most suitable virtual structure should be selected by the user considering properties such as diameter, bisection width and scalability. For example, the virtual structure can be a Ring of Rings (RoR), a hypercube, a CCC (Cube Connected Circle), or a torus.

Without loss of generality all results reported in this paper were obtained using the RoR structure [26]. The RoR is formed as follows: consider two integers, x and y, such that, $x \cdot y = n$, and let s be an integer such that $1 < s \leq y$. Set N is partitioned into x rings, called $N_0, N_1, \cdots, N_{x-1}$, in which, for each $a \in [0,x), N_a = \{i : a \cdot y \leq i < (a+1) \cdot y\}$. Link (i, j) belongs to E iff either $j \equiv (i+d)$ *mod* y for some $1 \leq d < s$ or $j \equiv (i+y)$ *mod* n. A notable feature of RoR structure is the high redundancy of virtual connections between nodes, whose degree is determined by parameters x, y, and s, i.e. there are several virtual connections available from any source to any destination.

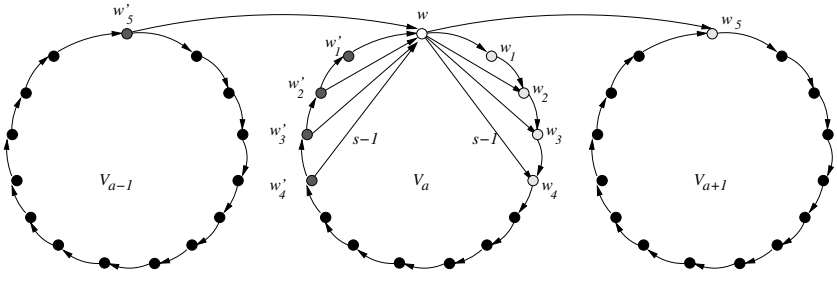


Fig. 1. RoR virtual structure with 3 rings and 15 nodes per ring

Figure 1 exemplifies a Ring of Rings (RoR) structure with 45 nodes ($n = 45$), divided in 3 rings ($x = 3$) with 15 nodes ($y = 15$) each. Further, each node has a direct connection to other five nodes ($s = 5$), meaning that they are responsible to collect and maintain the public keys of these five nodes. Moreover, nodes issue certificates binding the public-key to their respective nodes. In figure 1, for example, node w participates on the certificate issue procedure which binds k_{w_1} to w_1, k_{w_2} to w_2 and so on, and nodes w'_1, w'_2, w'_3, w'_4 and w'_5 mutually issue a certificate binding k_w to w. Note that nodes might not issue certificates to other nodes if they suspect that the other ones are misbehaving.

In *e*-VKM, each node i creates its own pair of public and private keys, pk_i and sk_i. After that, it must collaboratively issue certificates to other nodes following the virtual structure. Nodes only issue certificates to their virtual neighbors. A pair of nodes in the virtual structure must exchange their keys through a secure channel, i.e during a physical encounter over an infrared channel or via smart-cards or even using a key agreement protocol. All certificates are issued with a limited lifetime T_v, and after T_v, the certificate is considered expired. Before the certificate expiration, the issuers can update the certificate, issuing a new version with an extended T'_v. When a certificate is issued, its issuers store it in their local repository and send it to the correspondent node, which also stores the certificate. In the initial phase, nodes store only certificates which they issued and the certificates that were issued to them.

Certificate revocation can be done either explicitly or implicitly. Similar to PGP-Like, the implicit revocation is based on T_v. If the issuer does not update the certificate after T_v, the certificate is considered revoked. On the explicit revocation, the issuers revoke the certificate if they detect any misbehavior from the certificate owner.

When a node j wants to authenticate the public key of node i, it must use the virtual structure to discover which nodes had issued certificates to i. Then node j must

reactively validate the certificate with their respective issuers, e.g. nodes k_1, k_2 and k_3. If necessary, node j must also authenticate the public key of nodes k_1, k_2 and k_3 on the same way, i.e. selecting and validating certificates following the virtual structure. This process can be repeated until node j finds certificates which it considers to be valid or certificates issued by itself. Note that this behavior ensures that only correct and valid certificates will be used. However, as nodes must request certificates following the virtual structure before authenticating the key, it implies in a latency for authentications.

4.2 Multi-signature

e-VKM operations are based on the multi-signature scheme presented in [17]. Note that with some adjustments, any traceable and verifiable multi-signature scheme can be applied. The multi-signature is composed by three algorithms: (*i*) key generation, which outputs the public and private key of a node, based on global system information; (*ii*) multi-signature generation, an interactive protocol run by the virtual neighbors of a node to collaboratively issue its certificate; (*iii*) deterministic verification, which validates the signature of the presented certificate.

The key generation is based on a Gap Diffie-Hellman group of prime order. Public and private keys are extracted from the group. The issuing, revocation and update operations are performed using the multi-signature algorithm. Key authentication and certificate validation are performed via a deterministic verification algorithm. The verification algorithm solves a Decisional Diffie-Hellman problem in the group. Interested readers can check [17] for details about the Decisional Diffie-Hellman problem.

The advantages of the multi-signature are: (*i*) each member of the group can sign the message, while the size of the subgroup can be arbitrary [17]; (*ii*) it does not require a group manager, differently from group signatures [27]; (*iii*) it requires all nodes from a subgroup to sign the message, not allowing a single node to sign the message on behalf of the subgroup, as in ring signatures [28].

4.3 *e*-VKM Operations

The main operations of *e*-VKM are the creation of keys, issuing public key certificates, the authentication, update and revocation of certificates. The notation presented in Table 2 is assumed in all operations.

Creation of Keys. When joining the system, each node i receives the global system information (*GSI*), which contains all parameters and functions required to perform the cryptographic operations. Nodes might gather the *GSI* from the external entity before joining the system. The *GSI* contains the following information:

- g: a generator of the Gap Diffie-Hellman (GDH) group;
- p: a prime number used as the GDH order;
- $\mathcal{H}(\cdot)$: a hash function mapping arbitrary strings to the elements of $G \setminus \{1\}$, in which 1 denotes the identity element of G;
- $\mathcal{F}(\cdot)$: the virtual structure formation rule;

Recalling that each node i creates its own pair of public and private keys (pk_i and sk_i) in a self-organized and autonomous manner. In *e*-VKM, keys are created using Gap Diffie-Hellman groups obtained from bilinear maps. Let G be a Gap Diffie-Hellman group of prime order p and let g be a generator of the group. Thus, the secret key sk_i is a random element $x \in Z_p^* : sk_i = x$, and the public key pk_i is $y = g^x : pk_i = y$.

Issuing Certificates. After creating its key pair, each node i requests to its virtual neighbors a certificate binding its public key pk_i with its identity i. Before issuing a certificate to node i, each virtual neighbor must securely get the public key of i, to assure its authenticity. Nodes have many options to exchange their keys: (*i*) before network formation, when nodes are initializing the network; (*ii*) via a secure channel, as infrared, smart-cards or pendrives during a physical encounter of the users; (*iii*) using key agreement scheme to build a secure session. Even though key agreements are expensive solutions, nodes perform them only once, not excessively consuming network resources.

The set of virtual neighbors of node i, denoted by $VN(i)$ with s members, issues a unique certificate, following a traceable and verifiable multi-signature scheme [17]. Thus, all $j \in VN(i)$ sign the same certificate that will be used by node i in all secure network operations. To issue a signed certificate $C^i_{VN(i)}$, node i initially requests the certificate to its virtual neighbors by sending a *reqCert* message to them. Let $VN(i) = \{vn_i^1, vn_i^2, ..., vn_i^s\}$ be the set of s virtual neighbors of node i. Any node $j \in VN(i)$, upon receiving the *reqCert* message, takes the certificate C^i of node i, which has not been signed yet, and computes a signature $\sigma_j \leftarrow H(C^i)^{sk_j}$. Then, it sends the certificate C^i, with the signature σ_j and its own public key pk_j back to node i. As node i knows the virtual structure, it knows which nodes must send the certificate and the signature to it. Upon receiving all messages, node i computes $\sigma = \prod_{j \in VN(i)}(\sigma_j)$ and outputs $C^i_{VN(i)} = (C^i \| VN(i) \| \sigma)$. Then, it sends the signed certificate to all nodes in $VN(i)$. Thus, each node initially maintains its own certificate and the certificates it had partially issued.

Certificate Validation. When node k wants to authenticate the certificate $C^i_{VN(i)}$ of node i, it must verify the multi-signature of the issuers of the certificate, i.e all nodes in $VN(i)$. Node k takes the certificate $C^i_{VN(i)} = (C^i \| VN(i) \| \sigma)$ and extracts the list of public keys of $VN(i) = (pk_{i^1}, pk_{i^2}, ..., pk_{i^s})$ in which $pk_{i^j} = g^{x_j} = g^{sk_j}, \forall i^j \in VN(i)$. Then, node k computes $pk_{VN(i)} = \prod_{j \in VN(i)}(pk_j) = \prod_{j \in VN(i)} g^{x_j}$. The $pk_{VN(i)}$ represents the public key of the certificate issuers. After computing $pk_{VN(i)}$, node k must verify the multi-signature of the certificate using $\mathcal{H}_{\mathcal{DDH}} = (g, pk_{VN(i)}, H(C^i_{VN(i)}), \sigma)$. If $\mathcal{H}_{\mathcal{DDH}}$ is equal to 1 (one), then the signature is authentic and the certificate can be considered valid.

Certificate Update. The update or renewing of a certificate is similar to the issue process. If node i wants to update its certificate, it requests a new version of the certificate to its virtual neighbors. Every node $j \in VN(i)$ takes the certificate C^i of node i, with a new expiration date, and recomputes the signature $\sigma_j \leftarrow H(C^i)^{sk_j}$. Then, it sends C^i and σ_j to node i. Upon receiving all messages, node i recomputes $\sigma = \prod_{j \in VN(i)}(\sigma_j)$ and outputs $C^i_{VN(i)} = (C^i \| VN(i) \| \sigma)$, with the new expiration date.

Certificate Revocation. *e*-VKM supports two types of certificate revocation: implicit and explicit ones. The implicit revocation is based on the expiration time of the certificate and requires no extra communication. Upon reaching the expiration time of a certificate $C^i_{VN(i)}$, it is considered revoked.

The explicit revocation is invoked if node $j \in VN(i)$ wants to revoke the certificate $C^i_{VN(i)}$. To do so, it creates a signed revocation request message and sends it to all members of $VN(i)$. Upon receiving a revocation request, node $l \in VN(i)$ can agree or not with the revocation. If it does not agree, it just discards the message. If it agrees, i.e. it also wants to revoke the certificate, it creates a certificate RC^i of node i with a revocation flag, and computes a signature $\sigma_l \leftarrow H(RC^i)^{sl_l}$. Then, it sends the revoked certificate RC^i, with the signature σ_l and its own public key pk_l to the requesting node j. When node j receives all responses, it computes $\sigma = \prod_{l \in VN(i)}(\sigma_l)$ and outputs the revoked certificate $RC^i_{VN(i)} = (RC^i \| VN(i) \| \sigma)$. Finally, node j stores the revoked certificate in a local Certificate Revocation List (CRL) and sends it to all nodes of $VN_(i)$.

5 Evaluation

e-VKM was evaluated through simulations on the NS-2 v2.34. Table 3 summarizes the parameters used in the simulations. Results are averages of 35 simulations with 95% of confidence interval. Due to the similarity on the obtained results, the ones reported in this article are the worst case scenarios: 100 nodes, 120 meters of transmission range and maximum speed of 20m/s.

Table 3. Simulation scenarios

Parameter	Value
Network dimension	1000 x 1000 meters
Medium Access Control	IEEE 802.11 DCF
Transmission range	120 and 250 meters
Radio propagation model	two-ray ground
Nodes	50, 75 and 100
Mobility model	random waypoint
Max. speed	2 m/s up to 20 m/s
Max. pause time	20 seconds

In the PGP-Like, certificate exchanges are performed each 60 seconds and 600 random certificates are issued between nodes. *e*-VKM uses a RoR virtual structure with 4 rings, 25 nodes per ring and 5 virtual neighbors per node. These parameters are the same as the ones considered in other simulations, such as [6, 12, 15], to evaluate either the PGP-Like or the original VKM. Two metrics are used in the evaluation: Compromised Authentications(CA), and Compromised key Revocations (CR). They are defined as follows:

– **CA** is the rate of non-compromised nodes which can authenticate a false identity, created by a Sybil node. It represents how effective an attack is against the key management scheme. *CA* is denoted as follows:

$$CA = \frac{\sum_{i \in X} |i \rightsquigarrow f|}{|X|} \quad \forall f \in F \tag{1}$$

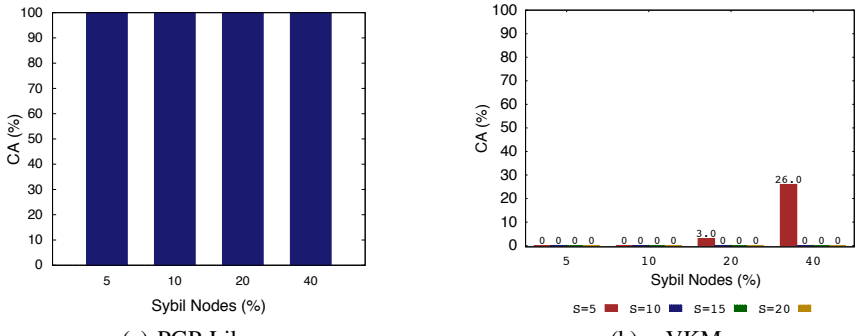
– **CR** is the rate of compromised certificate revocations performed by malicious nodes during a bad mouthing attack. It represents how effective the key management scheme is against such an attack. *CR* is denoted as follows:

$$CR = \frac{\sum_{i \in X} |f \overset{R}{\rightsquigarrow} i|}{|X|} \quad \forall f \in F \tag{2}$$

5.1 Sybil Attacks

Due to the fact that *e*-VKM uses the virtual structures in all authentications, it is very important to forbid the participation of a false identity, impersonated or stolen, into such a structure. Note that in *e*-VKM an attacker must compromise the virtual structure to inject a false identity into the system. Further, as the virtual structure is widely known, the creation of a new false identity into the virtual structure is not possible. In order to inject the false identity in the virtual structure, the attacker must change the virtual structure of all nodes at the same time.

Even though a false identity could not be fabricated, it might be stolen from an honest and valid node *i*, i.e. an attacker can impersonate a valid node. In this case, a malicious node must convince all *s* virtual neighbors from node *i* to issue a certificate to the false one. Thus, the only way a malicious node can inject an impersonate identity into the system is by acting in collusion with all virtual neighbors of node *i*.



(a) PGP-Like (b) *e*-VKM

Fig. 2. Compromised Authentications under Sybil Attacks

e-VKM was evaluated under Sybil attacks considering the metric *CA*, i.e the percentage of honest nodes that might authenticate a false identity. Results can be compared with the PGP-Like ones. Figure 2(a) shows that the Sybil attack totally affects the authentication in PGP-Like. In all scenarios, even under 5% of malicious nodes, 100% of

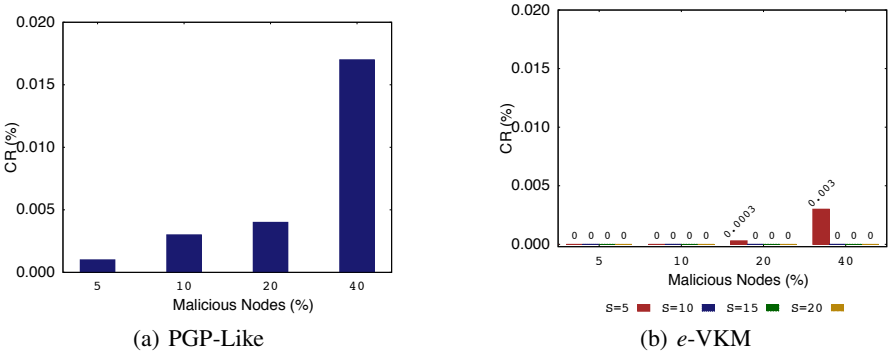
the false identities are authenticated by honest nodes. This happens because PGP-Like implements a transitive trust method and it has no identity check mechanism. Moreover, in the PGP-Like a malicious node does not need to convince other nodes to issue certificates to a false identity. Thus, if a malicious node creates a false identity and issues a certificate to it, all nodes that rely on such a malicious node will correctly authenticate the false identity.

The metric *CA* is also used to evaluate the impact of the Sybil attack on *e*-VKM authentications. Figure 2(b) shows that *e*-VKM is highly resistant to this attack. Only in scenarios with more than 20% of attackers acting in collusion and with $s = 5$ the percentage of compromised authentication is greater than zero, reaching a top of 3%. In scenarios with 40% of Sybil nodes and with $s = 5$, the amount of compromised authentications reaches 26%. Recalling that s is the number of virtual neighbors. In scenarios with s equals to 10, 15 or 20, *e*-VKM is completely resistant to Sybil attacks. Furthermore, no false identity is able to be authenticated in the system. These results show that, even under a high number of attackers, they are not able to impersonate a correct identity and issue a valid certificate to it.

It is important to point out that *e*-VKM might use reputation and/or misbehavior detection mechanisms to improve its resistance against misbehavior attacks. However, none of these improvements have been considered in the presented results.

5.2 Bad Mouthing Attacks

Finally, *e*-VKM was also evaluated under bad mouthing attacks. In this attack, malicious nodes produce false accusations against an honest node aiming to denigrate it. In the *e*-VKM, if the virtual neighbors of a node *i* are malicious or compromised by a bad mouthing attack, they might revoke the certificate issued to node *i*. In order to demonstrate *e*-VKM resistance to bad mouthing attacks, simulations of this attack against PGP-Like have also been performed. In PGP-Like, a bad mouthing attack is effective if malicious nodes can isolate an honest one, revoking all certificates issued to it.



(a) PGP-Like (b) *e*-VKM

Fig. 3. Compromised Revocations under Bad Mouthing Attacks

Figure 3(a) shows that the PGP-Like is highly resistant to these attacks. In all scenarios, even under 40% of malicious nodes, 0.02% of compromised revocations are

performed. This is due to the fact that PGP-Like allows nodes to issue several certificates to a unique node. Figure 3(b) shows the extreme resistance of *e*-VKM to the bad mouthing attack. In this case, the worst scenario for *e*-VKM happens with 40% of attackers and $s = 5$, and the percentage of compromised revocations is only 0.003%.

5.3 Scalability, Communication overhead and Latency

As stated, the virtual structure of *e*-VKM is static and well known by all nodes. This characteristic limits number of nodes in the system by the size of the virtual structure. Even though this is not desirable for some applications, in other ones it can be suitable, such as meeting communications, sensor networks and disaster rescue operations, in which parties are well known at network formation. Note that *e*-VKM allows nodes to leave the system at any time, affecting only the amount of virtual neighbors to issue and validate a certificate. It is also possible to create a very large virtual structure to accommodate the inclusion of several nodes in the system, as long as the connectivity of the virtual structure is guaranteed. The virtual structure can also be changed, augmented or reshuffled [26], though these operations are out of the scope of this paper.

The virtual structure construction has no overhead, as it is made through a simple formation equation (directly dependent on the virtual structure used), which is preloaded on the nodes by the external entity. The virtual structure maintenance has no overhead either as long as it is not augmented or reshuffled.

In terms of communication overhead, the *e*-VKM depends on the routing protocol used by the network. The worst case overhead to reactively validate a certificate is $s \times \Gamma + t$, in which Γ is the overhead of the routing protocol to build a route to a node. In other words, a node must contact all s issuers of a certificate and receive at least t replies in order to validate it. Moreover, the s certificates of the issuers might also be validated as well, and this process can be repeated several times until the origin finds certificates issued by itself. The worst case is $\kappa \times (s \times \Gamma + t)$, where κ is the diameter of the virtual structure. The latency of the protocol also depends on the routing protocol. The latency worst case is $s \times \gamma + \theta$, in which γ is the average latency to build a route by the used routing protocol, and θ is the largest round trip time between any chosen route. As the s certificates might also need to be validated and this process can be repeated several times until the origin finds certificates issued by itself, the worst case is $\kappa \times (s \times \gamma + \theta)$.

6 Conclusion and Future Work

Key management is a critical service in wireless ad hoc networks. It must deal with all security issues in a self-organized and decentralized way while considering nodes mobility and dynamic topology. Several application scenarios in WANETs, such as sensor networks, battlefield operations, health care solutions, or conference meetings, allow nodes to be loaded with critical information before joining the network. In these scenarios, the key management service may assume the existence of an external entity at the system initialization. Considering these scenarios, this paper has presented the enhanced VKM (*e*-VKM) which is extremely resistant to Sybil and bad mouthing attacks.

e-VKM uses a virtualization layer to indicate the certificate issue mechanism between nodes. Public and private keys are extracted from the Gap Diffie-Hellman group

of prime order. Certificates are issued using a traceable and verifiable multi-signature scheme. The issuing, revocation and update operations are performed using the multi-signature algorithm. Through multi-signature, *e*-VKM does not depend on group managers, and it does not allow a single node to sign the message on behalf of the group.

e-VKM was evaluated under Sybil and bad mouthing attacks. Results show that it is highly resistant to such attacks, keeping its effectiveness and performance even under 40% of attackers, while other schemes, e.g. PGP-Like, are completely vulnerable to the Sybil one. Future work includes the test of *e*-VKM under different kinds of attacks. It also includes the study on how to use *e*-VKM as the key management scheme for a secure virtualization-based routing protocol for WANETs.

Acknowledgments. This work is partially supported by CNPq/Brazil.

References

1. Zhang, C., Song, Y., Fang, Y.: Modeling secure connectivity of self-organized wireless ad hoc networks. In: Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM 2008), pp. 251–255. IEEE Communications Society (2008)
2. Djenouri, D., Khelladi, L., Badache, N.: A survey of security issues in mobile ad hoc and sensor networks. IEEE Surveys & Tutorials 7(4), 2–28 (2005)
3. Menezes, A.J., Oorschot, P.C.V., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Danvers (1996)
4. van der Merwe, J., Dawoud, D., McDonald, S.: A survey on peer-to-peer key management for mobile ad hoc networks. ACM Computing Survey 39(1), 1 (2007)
5. Khalili, A., Katz, J., Arbaugh, W.A.: Toward secure key distribution in truly ad-hoc networks. In: Proceedings of the 2003 Symposium on Applications and the Internet Workshops (SAINT 2003 Workshops), p. 342. IEEE Computer Society (2003)
6. Čapkun, S., Buttyán, L., Hubaux, J.P.: Self-organized public-key management for mobile ad hoc networks. IEEE Transactions on Mobile Computing 2(1), 52–64 (2003)
7. Čapkun, S., Hubaux, J.P., Buttyán, L.: Mobility helps peer-to-peer security. IEEE Transactions on Mobile Computing 5(1), 43–51 (2006)
8. Hubaux, J.P., Buttyán, L., Čapkun, S.: The quest for security in mobile ad hoc networks. In: Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc 2001), pp. 146–155 (2001)
9. Ngai, E.C.H., Lyu, M.R.: Trust- and clustering-based authentication services in mobile ad hoc networks. In: Proceedings of the 24th International Conference on Distributed Computing Systems Workshops (ICDCSW 2004), pp. 582–587. IEEE Computer Society (2004)
10. Ngai, E.C.H., Lyu, M.R., Chin, R.T.: An authentication service against dishonest users in mobile ad hoc networks. In: Aerospace Conference 2004, vol. 02, pp. 1275–1285. IEEE (2004)
11. Čapkun, S., Hubaux, J.P., Buttyán, L.: Mobility helps security in ad hoc networks. In: MobiHoc 2003: Proceedings of the 4th ACM International Symposium on Mobile ad hoc Networking & Computing, pp. 46–56. ACM Press (2003)
12. e Silva, R.F., da Silva, E., Albini, L.C.P.: Resisting impersonation attacks in chaining-based public-key management on manets: the virtual public key management. In: Proceedings of the International Conference on Security and Cryptography (SECRYPT 2009), pp. 155–158. INSTICC (2009)

13. Nogueira, M., Pujolle, G., da Silva, E., dos Santos, A., Albini, L.C.P.: Survivable keying for wireless ad hoc networks. In: Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM 2009), pp. 606–613. IEEE Communications Society (2009)
14. Zimmermann, P.R.: The official PGP user's guide. MIT Press, Cambridge (1995)
15. da Silva, E., Lima, M.N., dos Santos, A.L., Albini, L.C.P.: Quantifying misbehaviour attacks against the self-organized public key management on manets. In: Proceedings of the International Conference on Security and Cryptography (SECRYPT 2008), pp. 128–135. INSTCC Press, Porto (2008)
16. da Silva, E., Lima, M.N., dos Santos, A.L., Albini, L.C.P.: Analyzing the Effectiveness of the Self-organized Public-Key Management System on MANETs under the Lack of Cooperation and the Impersonation Attacks. CCIS, vol. 48, pp. 166–179. Springer, Heidelberg (2009)
17. Boldyreva, A.: Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2002)
18. Douceur, J.R.: The Sybil Attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
19. Dellarocas, C.: Mechanisms for coping with unfair ratings and discriminatory behavior in online reputation reporting systems. In: Proceedings of the 21th International Conference on Information Systems (ICIS 2000), pp. 520–525. Association for Information Systems, Atlanta (2000)
20. Michiardi, P., Molva, R.: Ad hoc networks security. ST Journal of System Research 4(1) (March 2003)
21. da Silva, E., Lima, M.N., dos Santos, A.L., Albini, L.C.P.: Identity-based key management in mobile ad hoc networks: techniques and applications. IEEE Wireless Communications Magazine 15 (2008)
22. Piro, C., Shields, C., Levine, B.N.: Detecting the Sybil attack in ad hoc networks. In: Proceeding of the IEEE/ACM International Conference on Security and Privacy in Communication Networks (SecureComm 2006), pp. 1–11. ACM (Augut 2006)
23. Wang, S.J., Tsai, Y.R., Chen, C.W.: Strategies averting Sybil-type attacks based on the Blom-scheme in ad hoc sensor networks. Journal of Communications (JCM) 3(1), 20–26 (2008)
24. Zhang, Q., Wang, P., Reeves, D.S., Ning, P.: Defending against sybil attacks in sensor networks. In: Proceedings of the Second International Workshop on Security in Distributed Computing Systems (SDCS) (ICDCSW 2005), pp. 185–191. IEEE Computer Society, Washington, DC (2005)
25. Christianson, B.: Why isn't trust transitive. In: Proceedings of the International Workshop on Security Protocols (WSP 1996). IEEE Computer Society (1996)
26. Albini, L.C.P., Caruso, A., Chessa, S., Maestrini, P.: Reliable routing in wireless ad hoc networks: the virtual routing protocol. Journal of Network and Systems Management 14(3), 335–358 (2006)
27. Chaum, D., van Heyst, E.: Group Signatures. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 257–265. Springer, Heidelberg (1991)
28. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)