Efficient Node Admission and Certificateless Secure Communication in Short-lived MANETs

Nitesh Saxena, *Member, IEEE*, Gene Tsudik, *Senior Member, IEEE*, Jeong Hyun Yi, *Member, IEEE*

***Abstract*— Decentralized node admission is an essential and fundamental security service in mobile ad hoc networks (MANETs). It is needed to securely cope with dynamic membership and topology as well as to bootstrap other important security primitives (such as key management) and services (such as secure routing) without the assistance of any centralized trusted authority. An ideal admission technique must involve minimal interaction among MANET nodes, since connectivity can be unstable. Also, since MANETs are often composed of weak or resource-limited devices, admission must be efficient in terms of computation and communication.**

Most previously proposed admission protocols are prohibitively expensive and require heavy interaction among MANET nodes. In this paper we focus on a common type of MANET that is formed on a temporary basis, and present a secure, efficient and a fully non-interactive admission technique geared for this type of a network. Our admission protocol is based on secret sharing techniques using bi-variate polynomials. We also present a new scheme that allows any pair of MANET nodes to efficiently establish an on-the-fly secure communication channel.

***Index Terms*— Security, distributed access control, authentication, cryptographic protocols, ad hoc networks, and mobile network protocols.**

Portions of this paper previously appeared in [46], [42].

Nitesh Saxena is with Computer and Information Science Department, Polytechnic University, USA. E-mail: nsaxena@duke.poly.edu. This work was done, in part, while at UCI.

Gene Tsudik is with School of Information and Computer Science, University of California, Irvine, USA. E-mail: gts@ics.uci.edu

Jeong Hyun Yi (corresponding author) is with Communication and Networking Lab., Samsung Advanced Institute of Technology, Korea. E-mail:jeong.yi@samsung.com. This work was done, in part, while at UCI.

This work is supported in part by an award from the Army Research Office (ARO) contract W911NF0410280, NSF awards 0331707 and 0331690 (ITR-RESCUE), and a grant from SUN Microsystems.

I. Introduction

MOBILE ad hoc networks (MANETs) have many well-known applications in military, law enforcement, emergency rescue and humanitarian aid environments. However, lack of stable infrastructure and absence of centralized control exacerbate security problems in MANETs thus requiring very specialized security services. *Admission Control* (or secure node admission) is a fundamental security service in MANETs. It is needed to ascertain membership eligibility and to bootstrap other important security services, such as secure routing [27], [26] and secure group communication [51], [50].

Secure node admission in MANETs cannot be performed centrally. This is because a centralized entity is a single point of failure which also represents an attractive and high-payoff attack target. Moreover, topology changes due to mobility and node outages may cause the central entity to be unreachable and thus unable to perform admission control for the entire network. This motivates us to investigate admission techniques that function in a distributed or decentralized manner. Since our emphasis is on security, the natural technology to consider is threshold cryptography.

The notion of threshold cryptography involves distributing cryptographic primitives (such as decryption or digital signatures) in order to secure them against corruption of a certain number of parties, called a threshold. For example, a (t, n) threshold signature scheme [14] allows a group of n parties to distribute the ability to digitally sign messages, such that any t parties can do so jointly, whereas, no coalition of less than t parties can sign. Such a threshold signature scheme is resilient against the so-called *static adversary* who corrupts at most $(t-1)$ parties in the entire lifetime of the system.

Two features of MANETs make decentralized node

admission a very challenging problem. *First*, MANET devices are often limited in terms of computational and battery power. *Second*, MANET nodes usually function in an asynchronous (on/off) manner, often becoming temporarily unavailable. Therefore, an ideal admission protocol must be efficient in terms of both computation and communication[1]. It must also involve minimal (ideally, *none* at all) interaction among nodes.

A number of admission techniques, which we discuss in the following section, have been proposed in recent years [30], [29], [33], [32], [36], [44], [45]. Most are based on (t, n) threshold cryptography and allow any set of t-out-of-n nodes (called sponsors) to admit a new node by giving it:

(1) a share of a group secret (to be used in future admissions), and
(2) a membership certificate (to be used for secure communication)

Unfortunately, all previous schemes are far from ideal. They all involve heavy sponsor interaction, in the process of either (1) or (2). Furthermore, they all are computationally expensive in performing (2). This severely limits their practicality.

Another common feature of prior techniques is the requirement that each new node be issued a certificate and a secret share, in a distributed manner. However, unlike wired networks, many MANETs are formed on a temporary basis. Examples include: a MANET formed for the duration of a conference program committee meeting (typically, one day), or a MANET formed by a group of rescuers in a disaster relief effort, as they remain in close proximity. We claim that such MANETs can benefit from much more efficient admission techniques, without sacrificing security. In particular, we observe that, in temporary MANETs, node admission can be realized by only issuing a node-specific secret share – (1) above – and thus obviating the need for expensive membership certificate issuance. This point is discussed in more detail in Section V.

Contributions: The contribution of this work is two-fold: First, we present a secure, efficient and fully non-interactive admission protocol for temporary MANETs. It is constructed using secret sharing techniques based on bi-variate polynomials. In contrast with prior work, our protocol eschews interaction and avoids any costly reliable broadcasts among admission sponsors. Second, we present a technique for setting up on-the-fly secure communication channels among MANET nodes. In particular, we show how to perform public key operations without any node certificates. This is achieved by using verifiable polynomial secret sharing as a key distribution scheme and treating secret shares as private keys. We thoroughly evaluate our proposal via real experiments and show that it compares very favorably to previous work.

[1]Communication is directly related to the consumption of battery power in MANET devices [1].

Organization: The rest of the paper is organized as follows: we first review prior work in Section II. Section III describes some preliminaries. The generic admission protocol is presented in Section IV, followed by the (inefficient) approach based on univariate polynomial secret sharing (UniAC) in Section V. We then describe, in Section VI, the admission protocol based on bi-variate polynomial secret sharing (BiAC) and accompanying techniques for establishing pairwise secure communication channels. Security arguments are presented in Section VII. Then, Section VIII, describes another realization of our proposal that uses identity-based cryptography. Finally, performance results are presented in Section IX.

II. Related Work

We now overview relevant prior work in MANET security. Zhou and Haas [55] first suggested the use of threshold cryptography for MANET security. The idea was to distribute trust among MANET nodes such that no less than a certain threshold is trusted. The key element of [55] is a distributed Certification Authority (CA) which issues certificates (using a threshold signature [14] protocol) to nodes joining the network. Although attractive, this idea is not directly applicable for MANET node admission. The approach is hierarchical in the sense that only select nodes can serve as components of the CA, i.e., take part in admission decisions. Moreover, contacting distributed CA nodes in a multi-hop and ever-changing MANET is not always possible.

Kong, et al. considered the same problem in series of papers [30], [29], [33], [32] and proposed a set ubiquitous and robust admission protocols. The security of these admission mechanism relies upon a special variant of the proactive threshold RSA signature scheme. Unfortunately, this scheme is neither robust [36] (i.e., it can not tolerate malicious nodes) nor secure [28]. We note that, thus far, all attempts to construct secure MANET admission protocols from secure threshold/proactive RSA signature schemes have failed [47].

Recently, Narasimha, et al. [36] and Saxena, et al. [45] proposed similar admission protocols based on

two flavors of discrete-logarithm based threshold signatures: threshold DSA [18] and threshold BLS [8], respectively. While provably secure, both solutions are inefficient. Of all known discrete-logarithm based threshold signature schemes, i.e., threshold-DSA [18], threshold-Schnorr [52], and threshold-BLS [8], only the last is non-interactive. However, the admission protocol in [45] that uses threshold BLS still requires some interaction.

To summarize, both heavy interaction and costly cryptographic computation make prior techniques overly expensive for many MANET applications.

In contrast, the admission technique developed in this paper is designed for short-lived MANETs and is completely non-interactive. It uses secret sharing based on so-called bi-variate polynomials which have been employed for related purposes in the literature [6], [35], [7]. In particular, [31] presents a key predistribution scheme for sensor networks using bivariate polynomials [7] *in the presence of a centralized authority*. The protocol we propose is fully distributed and allows MANET nodes to readily and efficiently share pairwise secret keys without any centralized support.

III. Preliminaries

A. Computation, Communication and Adversary Model

We operate in the standard model of threshold cryptography and distributed algorithms known as synchronous, reliable broadcast coupled with the static adversary [19]. This model involves nodes equipped with synchronized clocks[2]. We assume the existence of a naming system that pre-assigns each node with a unique identifier. We also assume that it is computationally hard for an adversary to forge identities.

We assume the existence of an on-line trusted public repository where the MANET-wide public key is stored. The nodes (both inside and outside the MANET) are connected by weakly synchronous communication network offering point-to-point channels and reliable broadcast[3]. To interact with a node in the network, an outsider must first retrieve the group public key from the repository.

[2]Clock synchronization is needed for system initialization with the distributed key generation protocol, such as [19]. The node admission protocol that we propose in this paper, on the other hand, does not require any synchrony

[3]The reliable broadcast channel is needed to ascertain the verifiability of shares distributed during the distributed key generation protocol, such as [19].

We consider the presence of the so-called "static" adversary, who at the beginning of system life-time (i.e., statically) schedules up to $t < n/2$ arbitrarily malicious faults among n MANET nodes. Such an adversary is said to "break" our scheme if it breaks the underlying node admission, key establishment, signature, or encryption schemes with respect to standard notions of security.

B. Discrete Logarithm Setting and Underlying Assumptions

In this paper, we work in the standard discrete logarithm setting: p, q are large primes s.t. q divides $p-1$ and g denotes a generator of subgroup G_q of order q in $\mathbb{Z}_p^*$. The primary cryptographic assumptions our protocols are based upon are as follows.

Assumption 1 (DL: Discrete Logarithm): Informally, DL assumptions means that it is infeasible to compute $x \in Z_q$, given (p, q, g, g^x).

Assumption 2 (CDH: Computational Diffie-Hellman): Informally, CDH assumption means that it is infeasible to compute g^{xy}, for $x, y \in Z_q$, given (p, q, g, g^x, g^y).

C. Random Oracle Model

Some of our proofs of security are in the standard, so-called Random Oracle Model (ROM) [3]. Informally, this means that the hash functions that we use are treated as ideal random functions. Doing security analysis in the ROM model effectively means that our proofs will consider only such attacks on the cryptographic schemes we propose whose success does not change if the fixed hash function like MD5 or SHA in these schemes are replaced with truly random functions.

D. Verifiable Secret Sharing

We use Shamir's secret sharing scheme [49] which is based on polynomial interpolation. To distribute shares among n nodes, a trusted dealer chooses a large prime q, and selects a polynomial $f(z)$ over $\mathbb{Z}_q$ of degree $t-1$ such that $f(0) = x$. The dealer computes each node's share x_i such that $x_i = f(id_i) \mod q$, and securely transfers x_i to node P_i. Then, any group of t nodes who have their shares can recover the secret using the Lagrange interpolation formula: $x = f(0) = \sum_{i=1}^{t} x_i \lambda_i(0) \pmod q$ where $\lambda_i(z) = \prod_{j=1, j\neq i}^{t} \frac{z - id_j}{id_i - id_j} \pmod q$.

Intuitively, the secret sharing idea comes from the fact that to recover a $t-1$ degree polynomial,

one needs to know t points on the polynomial. The knowledge of less than t points can not be used to learn the polynomial.

The idea of *Verifiable Secret Sharing* (VSS) [16] allows nodes to validate the correctness of the received shares. VSS setup involves two large primes p and q, and an element $g \in \mathbb{Z}_p^*$ chosen in a way that q divides $p-1$ and g is an element of $\mathbb{Z}_p^*$ which has order q. The dealer computes commitment to the coefficients a_i $(i = 0, \cdots, t-1)$ of the secret sharing polynomial in the form of witnesses W_i $(i = 0, \cdots, t-1)$, such that $W_i = g^{a_i} \pmod p$, and publishes these W_i-s in some public domain (e.g., a directory server). The secret share x_i can be validated by checking that $g^{x_i} = \prod_{j=0}^{t-1} (W_j)^{{id_i}^j} \pmod p$.

E. Schnorr Signature Scheme

The private key is x, chosen at random in $\mathbb{Z}_q$. The public key is $y = g^x \pmod p$. A Schnorr's signature [48] on message m is computed as follows. The signer picks a one-time secret k at random in $\mathbb{Z}_q$, and then computes $s = k + cx \pmod q$, $c = H(m, r)$ and $r = g^k \pmod p$. Signature (c, s) can be publicly verified by computing $r = g^s y^{-c} \pmod p$ and then checking if $c = H(m, r)$.

F. ElGamal Encryption Scheme

We use a variant of ElGamal Encryption [15], called *Hashed* ElGamal [5]. For a private and public key pair (x, y), the encrypter chooses a random $r \in \mathbb{Z}_q$ and computes the ciphertext (c_1, c_2) where $c_1 = g^r \pmod p$ and $c_2 = m \oplus H({y_i}^r)$ ($\oplus$ denotes the bitwise XOR operator). The plaintext can be obtained by computing $c_2 \oplus H(c_1^{x_i})$ from the ciphertext (c_1, c_2).

IV. Generic Admission Protocol

Notation used in the rest of paper is summarized in Table I.

Distributed node admission can be generally described as follows. At some point in time, all current MANET nodes: $P_1, \cdots, P_n$ share a secret x in a distributed manner – each P_i has its own secret share x_i. The requirement is that any $t-1$ (where t is a security parameter) secret shares do not yield any information about the secret x, whereas, any t secret shares completely define x. Now, a new node P_{n+1} needs to be admitted to the group and existing nodes need to supply P_{n+1} with its secret share x_{n+1} such that the new set: $x_1, x_2, \cdots, x_n, x_{n+1}$ satisfies the same requirement.

TABLE I
NOTATION USED IN THE REST OF THIS PAPER.

P_i	network node i
id_i	identity for P_i
t	admission threshold
n	total number of network nodes
$\mathbb{G}$	cyclic group in finite fields
g	generator of group $\mathbb{G}$
H	hash function such as SHA-1 or MD5
x_i	secret share of P_i
$x_i^{(j)}$	partial share for P_i by P_j
SL_i	list of sponsors for P_i
PK_i	temporary public key of P_i
$S_i(m)$	P_i's signature on message m
K_{ij}	pairwise key between P_i and P_j
$E_{K_{ij}}$	encryption with K_{ij}

Definition 1 (Node Admission Protocol): Let Ω be the set of current nodes $\{P_1, \cdots, P_n\}$, where each P_i has x_i, as above. Let $\Gamma \subset \Omega$ be any subset of t nodes and P_{n+1} is a new node. The process by which P_{n+1} acquires its secret share x_{n+1} from Γ is called a *(distributed) node admission protocol*. This protocol needs to have three properties:

1) *Correctness*: if all nodes follow the protocol correctly, P_{n+1} successfully acquires x_{n+1}, and the new set of secret shares $x_1, x_2, \cdots, x_n, x_{n+1}$ satisfies the same requirement: any subset of $t-1$ secret shares does not yield any information about x, while any subset of t secret shares yields x.
2) *Security*: the protocol does not leak any information about neither the secret share of any existing node (that takes part in the admission protocol) nor the secret x, even to an adversary who has corrupted at most $t-1$ existing nodes.
3) *Robustness*: if any malicious nodes that participate in the protocol try to disrupt the protocol by providing incorrect values, the new node can detect them.

The basic operations of a generic admission protocol are composed of the following steps:

1) *Bootstrapping*: The group is initialized by either a trusted dealer or a set of founding nodes. The dealer or a set of founding nodes chooses the group secret key, then computes and publishes the corresponding public parameters. The group secret is shared among the initial member nodes using either Shamir's secret sharing [49] or Joint Secret Sharing (JSS) [19] techniques. The share of the group secret held by each node is called its *secret share*.

2) *Node Admission*: A prospective new node P_{n+1} must be issued its secret share by current nodes. Fig-

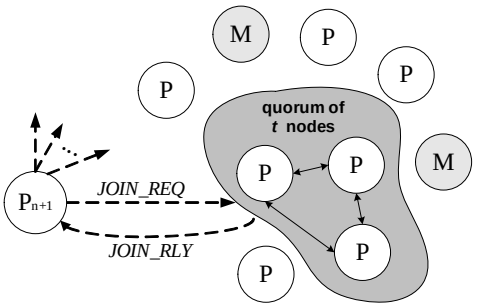


Fig. 1. Overview of Node Admission Protocol. P and M indicates (honest) network node and malicious node, respectively.

ure 1 gives a high-level view of admission protocol. Note that, depending on the underlying cryptographic technique, this step may involve multiple rounds and/or co-ordination among nodes who commit to P_{n+1}.

- P_{n+1} initiates the admission protocol by sending a `JOIN_REQ` message to the group.
- A node that receives `JOIN_REQ` message and approves the admission of P_{n+1} replies, over a secure channel,[4] with a partial secret share for P_{n+1} derived from its own secret share.
- Once P_{n+1} receives partial shares from at least t different sponsors, it pools them together to compute its new secret share.
- Finally, to achieve robustness, P_{n+1} verifies its new secret share. (This is needed since a malicious sponsor can sabotage admission of P_{n+1} by providing incorrect partial secret shares, leading to a denial-of-service (DoS) attack.) This feature is called *verifiability*. Also, if P_{n+1} detects that its new secret share is invalid, it must be able to trace the bogus share(s) and the corresponding malicious sponsor(s). This functionality is provided by the so-called *traceability* feature. We note that verifiability is always required, whereas, traceability is only necessary if a new node detects that its freshly reconstructed secret is invalid.

Once admitted, P_{n+1} becomes a genuine MANET node (group member). Thereafter, the following operations are needed to enable secure communication among nodes.

3) *Pairwise Key Establishment*: This is needed to secure communication between any pair of nodes, e.g., as required by secure routing protocols, such as Ariadne [27].

4) *Signing*: This is required in cases when *non-repudiation* is needed, e.g., as in ARAN secure routing protocol [13], and in general, when a single message needs to be multicasted or broadcasted in an authenticated manner.

5) *Encryption*: This is needed to allow outside entities to communicate securely with MANET nodes. For example, in a MANET where nodes function as mobile sensors, a base station might want to issue a confidential query a particular node (or a set thereof) to obtain measurements.

V. UniAC: Previous Methods

As mentioned earlier, unlike wired networks, many MANETs involve temporary operation and can benefit from more efficient admission techniques, without sacrificing security. In such settings, we can consider only a static adversary and, therefore, secret shares need not to be periodically updated (as in so-called proactive update protocols, e.g., [24]). Therefore, the secret sharing polynomial *remains constant* throughout the lifetime of the MANET and the commitment to this polynomial can act as the group public key. The commitment to each node's secret share is derivable from (and thus automatically bound to) the group public key. Therefore, node-specific membership certificates are not needed. Nodes can use their secret shares (and/or the group public key) to secure communication among themselves.

Previous MANET admission techniques [30], [29], [33], [36], [44], [45] constructed using the mobile adversary model can be adapted for our purposes by removing the un-needed certificate issuance procedure. In this section, we briefly describe how to adapt these protocols. Since these protocols are all based on uni-variate polynomial secret sharing, we refer to them collectively as: *Uni*Variate *A*dmission *C*ontrol or UniAC.

A. Bootstrapping

The system can be initialized by a trusted dealer *TD* or a set of founding nodes. As in Shamir's secret

[4]One way to set up a secure (secret and authenticated) channel between P_{n+1} and each sponsor is with device pairing techniques based on out-of-band (OOB)channels [34], [23], [43], [53]. Alternatively, if P_{n+1} and each sponsor have a common trusted CA, a secure channel can be trivially established using any secure authenticated key agreement protocol, e.g., [54]. Our admission protocol allows P_{n+1} to establish secure channels with any node, once it establishes secure channels with any a subset of t nodes. Since all communication between P_{n+1} and sponsors in the admission protocol flows over secure channels, "man-in-the-middle" attacks are prevented.

sharing [49] based on a uni-variate polynomials, the *TD* (or founding nodes) choose(s) a large prime q, and select(s) a polynomial $f(z) = \sum_{i=0}^{t-1} a_i z^i \pmod q$ such that $f(0) = x$, where a_i-s are the coefficients of the polynomial, q is a large prime, and x is the group secret. The *TD* computes each node's secret share x_i such that $x_i = f(id_i) \pmod q$, and securely transfers x_i to node P_i. *TD* also publishes a commitment to the polynomial as in VSS.

B. Node Admission

During the admission protocol (see Figure 2), P_{n+1} is given the *shuffled* partial secret share as $\tilde{x}_{n+1}^{(j)} = x_j \lambda_j(id_{n+1}) + R_j \lambda_j(0)$ by a sponsoring node P_j. Upon receiving partial share values from t admitting nodes, P_{n+1} obtains its secret share x_{n+1} by simply summing up the partial shares, performing VSS and, if needed, the traceability procedure. (See [11] for details regarding the actual computation involved in these procedures.)

$P_{n+1} \rightarrow P_i$:	$id_{n+1}, PK_{n+1}, S_{n+1}(id_{n+1}, PK_{n+1})$	(1)
$P_{n+1} \leftarrow P_i$:	$id_i, PK_i, S_i(id_i, PK_i)$	(2)
$P_{n+1} \rightarrow P_j$:	$SL_{n+1}, S_{n+1}(SL_{n+1})$	(3)
$P_i \longleftrightarrow P_j$:	*Random Shuffling*	(4)
$P_{n+1} \leftarrow P_j$:	$E_{PK_{n+1}}\{\tilde{x}_{n+1}^{(j)}\}$	(5)

Fig. 2. UniAC Admission Protocol. To secure the protocol against *replay* attacks, appropriate nonces or timestamps and protocol message identifiers need to be included in each step. However, we omit these values to keep our description simple.

Note that, in order to compute Lagrange coefficients $\lambda_j(id_{n+1})$ in Step (5), t sponsors need to be aware of each other's id-s. Also, since $\lambda_j(id_{n+1})$-s are publicly known, P_{n+1} can derive x_j from $x_{n+1}^{(j)}$. This is prevented by using the Joint Zero Secret Sharing technique [47] which entails adding an extra random value R_j to each share. R_j-s are securely shared between sponsors P_i and P_j and sum up to zero, by construction. This process – called *Random Shuffling* – is illustrated in Figure 3(a).

Note that, due to $\lambda_j(id_{n+1})$ computation and Random Shuffling, the admission protocol involved *heavy interaction* among t sponsors: $O(t^2)$ point-to-point and $O(t)$ reliable broadcast messages [10]. This overhead is impractical for most MANETs.

C. Pairwise Key Establishment

Any pair of genuine nodes P_i, P_j can establish a shared secret key using their respective secret shares x_i, x_j and public VSS information. P_i computes the public key y_j of P_j (only knowing its identifier id_j) as $y_j = \prod_{l=0}^{t-1} (W_l)^{{id_j}^l} \pmod p$ from public witness values, and exponentiates the result with its share x_i to obtain key $k_{ij} = {y_j}^{x_i} = (g^{x_j})^{x_i} \pmod p$. Similarly, P_j computes $y_i = \prod_{k=0}^{t-1} (W_k)^{{id_i}^k} \pmod p$ and exponentiates it with x_j to obtain k_{ji}. Since $k_{ij} = k_{ji}$, P_i and P_j can use $K_{ij} = H(k_{ij}) = H(k_{ji})$, as a session key for secure communication.

This key establishment procedure is secure under the CDH assumption. In other words, an adversary who corrupts at most $(t-1)$ nodes can not compute a shared key between any pair of honest nodes (as long as the CDH assumption holds). A more detailed security argument can be found in [42].

D. Signing

To sign a message m, P_i (who has a secret share x_i) picks a random secret $k \in Z_q$ and computes $r = g^k \pmod p$. It then outputs the signature as a pair (c, s), where $c = H(m, r)$ and $s = k + cx_i \pmod q$. In order to verify the above signature (c, s), a recipient first computes P_i's public key $y_i = \prod_{l=0}^{t-1} (W_l)^{{id_i}^l} \pmod p$, and verifies whether $c = H(m, r)$, where $r = g^s {y_i}^{-c} \pmod p$. The security of this scheme is discussed in [42].

E. Encryption

To encrypt a message m for P_i, the encrypter computes P_i's public key $y_i = \prod_{j=0}^{t-1} (W_l)^{{id_i}^l} \pmod p$, chooses a random $r \in \mathbb{Z}_q$ and then sends a pair (c_1, c_2) to P_i, where $c_1 = g^r \pmod p$, $c_2 = m \oplus H({y_i}^r)$ and $\oplus$ denotes the bit-wise XOR operation. P_i decrypts by computing $c_2 \oplus H(c_1^{x_i})$ from ciphertext (c_1, c_2). Once again, the security of this scheme is analyzed in [42].

VI. BiAC: Non-Interactive Node Admission

We now propose a new non-interactive admission protocol based on secret sharing with bi-variate polynomials. We call it *Bi*Variate *A*dmission *C*ontrol or BiAC for short.

A. Overview

As shown in Figure 3(b), we avoid interaction among sponsors by using a *bi-variate* polynomial $f(z, y)$.

To distribute shares among n nodes, a trusted dealer *TD* chooses a large prime q and selects a random symmetric bi-variate polynomial $f(z, y) = \sum_{\alpha=0}^{t-1} \sum_{\beta=0}^{t-1} f_{\alpha\beta} z^\alpha y^\beta \pmod q$ such that $f(0, 0) = x$, where the constants $f_{\alpha\beta}$-s are the coefficients of the polynomial and x is the group secret. Since the

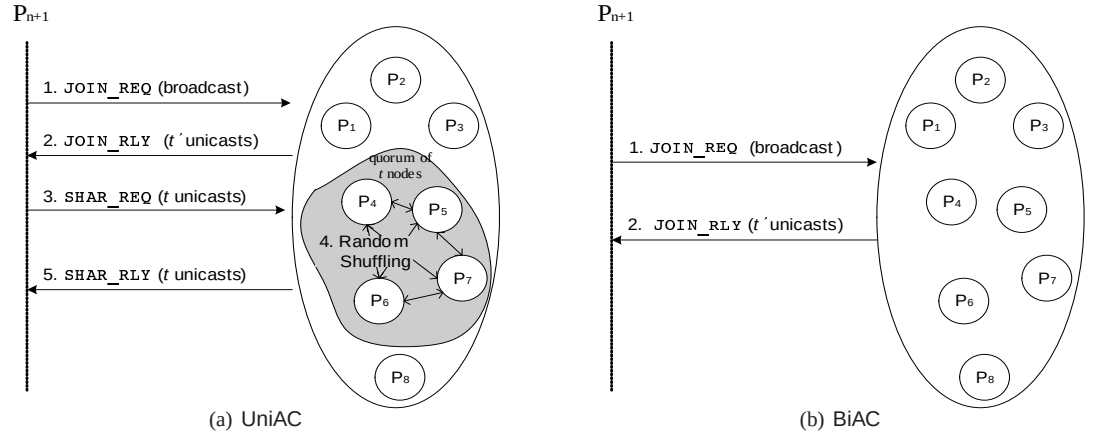


Fig. 3. Comparison of UniAC and BiAC. In UniAC, due to Lagrange interpolation, t sponsors need to be aware of each other's id-s and random shuffling is required. In contrast, BiAC requires neither.

polynomial is symmetric, $f_{\alpha\beta} = f_{\beta\alpha}$ for each α, β and $f(z, y) = f(y, z)$. For each node P_i, *TD* computes a uni-variate polynomial, called a *share polynomial*, $x_i(z)$ of degree $(t-1)$ such that $x_i(z) = f(z, id_i) \pmod q$, and securely transfers $x_i(z)$ to each node P_i. Note that, after initializing at least t nodes, the dealer is no longer needed.

In order to admit P_{n+1}, each sponsor must issue to it a *share-polynomial* $x_{n+1}(z)$ in a distributed manner. This is achievable if at least t sponsors supply P_{n+1} with partial shares $x_j(id_{n+1})$ such that $x_j(id_{n+1}) = f(id_{n+1}, id_j)$ for $j \in [1, n]$. P_{n+1} can then compute $f(id_{n+1}, z)$ (which is identical to $f(z, id_{n+1})$ since $f(z, y)$ is symmetric) and thus obtain its share-polynomial $x_{n+1}(z) = f(z, id_{n+1})$ from t partial shares $x_j(id_{n+1})$.

Unlike UniAC, this scheme *does not* require any interaction among sponsors.

B. Bootstrapping

The group is initialized by either a single node (centralized initialization) or a set of nodes (decentralized initialization).

1) Centralized Initialization. *TD* computes a two-dimensional sharing of the secret by choosing a random bi-variate polynomial:

$$f(z, y) = \sum_{\alpha=0}^{t-1} \sum_{\beta=0}^{t-1} f_{\alpha\beta} z^{\alpha} y^{\beta} \pmod q \qquad (1)$$

such that $f(0, 0) = x$, for the group secret x. *TD* computes $W_{\alpha\beta}$ $(\alpha, \beta \in [0, t-1])$, called witnesses:

$$W_{\alpha\beta} = g^{f_{\alpha\beta}} \pmod p \qquad (2)$$

and places $W_{\alpha\beta}$-s into a public repository. Once *TD* computes the witness matrix, it sends each P_i $(i \in [1, n])$ a distinct *share-polynomial:* $x_i(z) = f(z, id_i)$. *TD*'s presence is no longer needed after this initialization phase.

2)Decentralized Initialization. Assuming a set of t or more founding nodes. These t nodes agree on a random bi-variate polynomial $f(z, y)$ using the JSS protocol.

C. Node Admission

To join the group, P_{n+1} must collect at least t partial shares of the polynomial. Figure 4 shows the protocol messages.

$$\begin{array}{lll} P_{n+1} \rightarrow P_i\text{:} & id_{n+1}, PK_{n+1}, S_{n+1}(id_{n+1}, PK_{n+1}) & (1) \\ P_{n+1} \leftarrow P_i\text{:} & id_i, PK_i, E_{PK_{n+1}}\{x_i(id_{n+1})\}, & (2) \\ & S_i(id_i, PK_i, E_{PK_{n+1}}\{x_i(id_{n+1})\}) & \end{array}$$

Fig. 4. BiAC Admission Protocol (No interaction among P_i-s is required).

1) Same as the step (1) in Section V.

2) After verifying the signature of the JOIN_REQ message, each prospective sponsor P_i who wants to admit P_{n+1} computes a *partial share* $x_i(id_{n+1})$ using its own *share-polynomial*, such that: $x_i(id_{n+1}) = f(id_{n+1}, id_i)$.

P_i then replies to P_{n+1} with a JOIN_RLY message. Each JOIN_RLY is signed by the sender and contains encrypted $x_i(id_{n+1})$ along with: id_i and PK_i.

To compute their partial shares, sponsors do not need to be aware of each other which avoids interaction. This is unlike UniAC, where each sponsor needs

know about all other sponsors in order to compute the Lagrange coefficient in partial share issuance.

3) Upon receiving t' $(\geq t)$ `JOIN_RLY` messages, P_{n+1} selects *any* t of them and computes its own share-polynomial $x_{n+1}(z)$ using standard Gaussian elimination [41]. We denote the share-polynomial $x_{n+1}(z)$ reconstructed by P_{n+1} as $\sum_{\alpha=0}^{t-1} A_\alpha z^\alpha$. Since $x_i(id_{n+1}) = x_{n+1}(id_i)$, due to the symmetry, the selected t partial shares $\{x_{n+1}(id_1), \cdots, x_{n+1}(id_t)\}$ can be represented as

$$\begin{aligned}
A_0 + A_1 id_1 + A_2 {id_1}^2 + \cdots + A_{t-1} {id_1}^{t-1} &= x_{n+1}(id_1) \\
A_0 + A_1 id_2 + A_2 {id_2}^2 + \cdots + A_{t-1} {id_2}^{t-1} &= x_{n+1}(id_2) \\
&\vdots \\
A_0 + A_1 id_t + A_2 {id_t}^2 + \cdots + A_{t-1} {id_t}^{t-1} &= x_{n+1}(id_t)
\end{aligned}$$

Thus, the problem of interpolating $x_{n+1}(z)$ using t $x_i(id_{n+1})$-s is equivalent to the problem of computing a matrix A, such that $XA = B$:

$$\begin{bmatrix} (id_1)^0 & \cdots & (id_1)^{t-1} \\ (id_2)^0 & \cdots & (id_2)^{t-1} \\ \vdots & \cdots & \vdots \\ (id_t)^0 & \cdots & (id_t)^{t-1} \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ \vdots \\ A_{t-1} \end{bmatrix} = \begin{bmatrix} x_{n+1}(id_1) \\ x_{n+1}(id_2) \\ \vdots \\ x_{n+1}(id_t) \end{bmatrix}$$

The above system of linear equations yields a unique solution since id_i values are distinct and matrix $X = [x_{ij}]$, where $x_{ij} = (id_i)^{j-1}$ for all $i, j \in [0, t]$, is invertible.

In order to validate the acquired share-polynomial $x_{n+1}(z)$, P_{n+1} must perform the following verifiability procedure:

$$A_\alpha = \sum_{\beta=0}^{t-1} f_{\alpha\beta} (id_{n+1})^\beta \tag{3}$$

for $\alpha \in [0, t-1]$. Using the public witness values $W_{\alpha\beta} = g^{f_{\alpha\beta}} \pmod p$, the polynomial can be verified:

$$g^{A_\alpha} = \prod_{\beta=0}^{t-1} (W_{\alpha\beta})^{(id_{n+1})^\beta} \pmod p \tag{4}$$

for $\alpha \in [0, , t-1]$.

The right-hand side in this equation can be pre-computed by P_{n+1} prior to starting the admission process.

If verification fails, P_{n+1} must trace the faulty share(s) via the traceability procedure. This involves verifying the validity of each partial share $x_i(id_{n+1}) = f(id_{n+1}, id_i)$ as follows:

$$g^{x_i(id_{n+1})} = \prod_{\alpha=0}^{t-1} \prod_{\beta=0}^{t-1} (W_{\alpha\beta})^{(id_{n+1})^\alpha (id_i)^\beta} \pmod p \tag{5}$$

Note that $\prod_{\alpha=0}^{t-1} (W_{\alpha\beta})^{(id_{n+1})^\alpha}$ in the this equation can be pre-computed since $W_{\alpha\beta}$-s and id_{n+1} are known to P_{n+1} in advance.

D. Pairwise Key Establishment

Once every node has its share-polynomial, pairwise key establishment is the same as in [7] and [31]. Any pair of nodes P_i and P_j establish a shared key as follows: P_i uses its share-polynomial $f(z, id_i)$ to compute $K_{ij} = f(id_j, id_i)$. Similarly, P_j uses its $f(z, id_j)$ to compute $K_{ji} = f(id_i, id_j)$. Since $f(z, y)$ is symmetric, $K_{ij} = K_{ji}$. Thus, P_i and P_j share a secret key usable for subsequent secure communication.

E. Signing

In the context of BiAC, signing is very similar to that in UniAC. The only difference is that, when generating a signature, the secret key of P_i is $x_i = x_i(0) = f(0, id_i)$. P_i's public key $y_i = g^{x_i} \pmod p$ is computed using VSS witnesses and node identifier id_i as $y_i = \prod_{\beta=0}^{t-1} (W_{0\beta})^{{id_i}^\beta} \pmod p$. The actual signing is done using Schnorr's signature scheme; it is denoted as *BiAC-Sig*.

1) Signing.: To sign a message m , P_i (having a private key x_i), picks a random secret $k \in Z_q$ and computes $r = g^k \pmod p$. It then outputs the signature as a pair: (c, s), where $c = H(m, r)$ and $s = k + cx_i \pmod q$.

2) Verification.: To verify a signature (c, s), a recipient first computes the public key y_i of the signer P_i as $y_i = \prod_{\beta=0}^{t-1} (W_{0\beta})^{{id_i}^\beta} \pmod p$, and verifies whether $c = H(m, r)$, where $r = g^s {y_i}^{-c} \pmod p$.

Security of this scheme is discussed in Section VII below.

F. Encryption

We use the hashed ElGamal encryption scheme (described in Section III-F) and refer to it as *BiAC-Enc*.

1) Encryption.: To encrypt a message m for P_i, the encrypter first obtains P_i's public key $y_i = \prod_{\beta=0}^{t-1} (W_{0\beta})^{{id_i}^\beta} \pmod p$, picks a random $r \in \mathbb{Z}_q$ and finally computes $c_1 = g^r \pmod p$ and $c_2 = m \oplus H({y_i}^r)$. It then sends (c_1, c_2) to P_i.

2) Decryption.: P_i recovers m by computing $c_2 \oplus H(c_1^{x_i})$ from ciphertext (c_1, c_2).

The security argument of this scheme is discussed in the next Section.

VII. Security Arguments

We now discuss security arguments for BiAC node admission and pairwise key establishment.

A. BiAC Node Admission

As shown in Section VI-C, BiAC satisfies both "correctness" and "robustness" properties. Security of BiAC protocol is based on the DL assumption, as long as the adversary can not corrupt more than $(t-1)$ $(< n/2)$ nodes. We briefly sketch out this argument. Basically, as in the well-known Feldman's VSS [16], we use *simulated adversarial view* to show that an adversary, who corrupts at most $(t-1)$ nodes, learns nothing (other than the witness $g^x \pmod p$) about the secret x, during the initialization and admission procedures of the scheme. This is achieved by generating a simulator which, on input $g^x \pmod p$, produces public information and private information to the adversary which is statistically indistinguishable from the one produced in the actual execution of these procedures.

B. BiAC Pairwise Key Establishment

Unlike pairwise key establishment in UniAC (for which security is based on the CDH assumption), security of BiAC pairwise key establishment described in Section VI-D is unconditional, i.e., not based on any complexity assumption. Detailed security arguments for this claim can be found in [7].

C. BiAC Signing

We argue that *BiAC-Sig* remains secure against the standard notion of existential forgery under chosen message attack (CMA) [21] in ROM [3] as long as the DL assumption holds. Note that *BiAC-Sig* is different from regular signatures in the sense that: (1) nodes generate signatures with related (and not completely independent) secret keys, and (2) the adversary knows at most $t-1$ of these secret keys.

Theorem 1 (Security of *BiAC-Sig*): Under the DL assumption in ROM, as long as the adversary corrupts no more than $t-1$ nodes, *BiAC-Sig* is secure against the chosen-message attack for every remaining uncorrupted node

The proof of the above theorem can be found in Appendix I.

D. BiAC Encryption

We show that *BiAC-Enc* is secure with respect to the standard notion indistinguishability, as long as the DL assumption holds. Indistinguishability [20] is defined as the following game: the adversary, who is given the public key, outputs two challenge messages. Next, one of these messages is encrypted and given back to the adversary. The adversary is said to win the game if he can determine – with probability non-negligibly over 0.5 – which of the two messages was encrypted

The indistinguishability notion was originally geared for a single node scenario, where multiple messages are encrypted for that one node. To capture the security of *BiAC-Enc*(where we have multiple nodes and messages are encrypted using *related* keys) we adopt the *multi-node* indistinguishability notion of Baudron, et al. [2] and Bellare, et al. [4]. In this notion, the game is as follows: first the adversary is given n public keys $(PK_1, \cdots, PK_n)$ of all nodes. The adversary then outputs two vectors of n messages $M_0 = \{m_{01}, \cdots, m_{0n}\}$ and $M_1 = \{m_{11}, \cdots, m_{1n}\}$, which might be related or same, as challenges. One of the vectors M_b (b is 0 or 1) is then encrypted with n public keys (the order of the encryption is preserved, i.e., m_{bi} is encrypted with PK_i). The adversary wins the game if it succeeds – with probability non-negligibly over 0.5 – determine which message was encrypted. It has been shown in [4], [2] that an encryption scheme secure in the sense of single-node indistinguishability is also secure in the sense of multi-node indistinguishability.

Theorem 2 (Security of *BiAC-Enc*): Under the CDH assumption in ROM, as long as the adversary corrupts no more than $t-1$ nodes, *BiAC-Enc* is secure in terms of multi-node indistinguishability.

The proof of this theorem can be found in Appendix II.

VIII. Comparison with ID-based Cryptography

One interesting side-effect of the discussion in Sections VII-C and VII-D is that *BiAC-Sig* and *BiAC-Enc* can be viewed as identity-based signature and encryption schemes, respectively. Basically, a trusted center provides each node with a secret value (VSS share, in our case) derived from that node's unique identifier, and publishes the VSS information as its public key. Knowing the identifier of a particular node and the public key of the trusted center, one can send encrypted messages and verify signatures. This

is equivalent to IBE [9], and ID-based signatures [12], apart from the fact that our scheme becomes insecure if there are more than $t-1$ collusions or corruptions. However, unlike other ID-based schemes, our proposal is based on standard cryptographic assumptions. Moreover, for reasonable group sizes and threshold values, *BiAC-Enc* is much more efficient than other ID-based encryption schemes which require costly operations (e.g., scalar point multiplications, map-to-point operations and bilinear mappings [9]) on elliptic curves[5].

TABLE II
FEATURE COMPARISON

Key Features	UniAC	BiAC
Approach	ID-based	ID-based
Security Assumption (ADMIT)	DL	DL
Security Assumption (KEYEST)	CDH	**Unconditional**
Security Assumption (SIGN)	DL	DL
Security Assumption (ENCRYPT)	CDH	CDH
Minimum Network Size	$2t-1$	$2t-1$
Decentralized Admission	Yes	Yes
DoS Resistance (TRACE)	Yes	Yes
Interaction among Sponsors	Yes	**No**
Random Shuffling Required	Yes	**No**
Reliable Broadcast Required	Yes	**No**

ADMIT: node admission, TRACE: traceability, SIGN: signing
KEYEST: pairwise key establishment, ENCRYPT: encryption

IX. PERFORMANCE ANALYSIS

We now discuss the implementations of UniAC and BiAC for temporary MANETs and compare them in terms of node admission, traceability, pair-wise key establishment, signing and encryption costs. We also summarize and compare some salient features in Table II. As expected, BiAC significantly outperforms UniAC overall.

A. Complexity Analysis and Comparison

We summarize computation and communication complexities[6] in Table III. Note that computational cost is measured in the number of modular exponentiations – the most computationally intensive operation. Communication complexity reflects the costs of the admission protocol.

TABLE III
COMPLEXITY COMPARISON

Category			UniAC	BiAC
Comp. (# Exp)	ADMIT	P_{n+1}	1	t
		P_i	t	0
	TRACE		t^2+3t	t^2+t
	KEYEST		$t+1$	0
	SIGN	SIG	1	1
		VER	$t+2$	$t+2$
	ENCRYPT	ENC	$t+2$	$t+2$
		DEC	1	1
Comm. (ADMIT)	Rounds	broadcast	1	1
		unicast	t^2+2	t
	B/W	JOIN_REQ	$t\log q+ t\log p$	$t\log q+ t\log p$
		JOIN_RLY	$t\log q+ t\log p$	$2t\log q+ t\log p$
		SHAR_REQ	$t^2\log q$	N/A
		SHAR_RLY	$t^2\log q$	N/A

ADMIT: node admission, TRACE: traceability, KEYEST: pairwise key establishment

BiAC requires each sponsor P_i to perform $O(t)$ modular multiplications and P_{n+1} – $O(t^3)$ modular multiplications for Gaussian elimination and $O(t)$ exponentiations for verifiability. Whereas, UniAC entails each P_i performing $O(t)$ multiplications, and P_{n+1} – $O(t)$ multiplications plus one exponentiation for verifiability. For traceability, both the schemes require $O(t^2)$ multiplications and $O(t^2)$ exponentiations, with pre-computation. BiAC is significantly more efficient than UniAC for computing pairwise keys, since the former requires only $O(t)$ multiplications, while the latter needs $O(t)$ exponentiations as well as $O(t)$ multiplications. We note that pairwise key establishment is a very frequent operation in a MANET; thus, its efficiency is extremely important. For singing, both UniAC and BiAC require one exponentiation for signature generation and $O(t)$ – for signature verification. The encryption cost for both schemes follows same pattern; $O(t)$ exponentiations for encryption and one exponentiation for decryption.

As far as overall communication costs[7], BiAC consumes $O(t\log q)$ and $O(t\log p)$ bits, while UniAC – $O(t^2\log q)$ plus $O(t\log p)$ bits, due to the interactive random shuffling procedure.

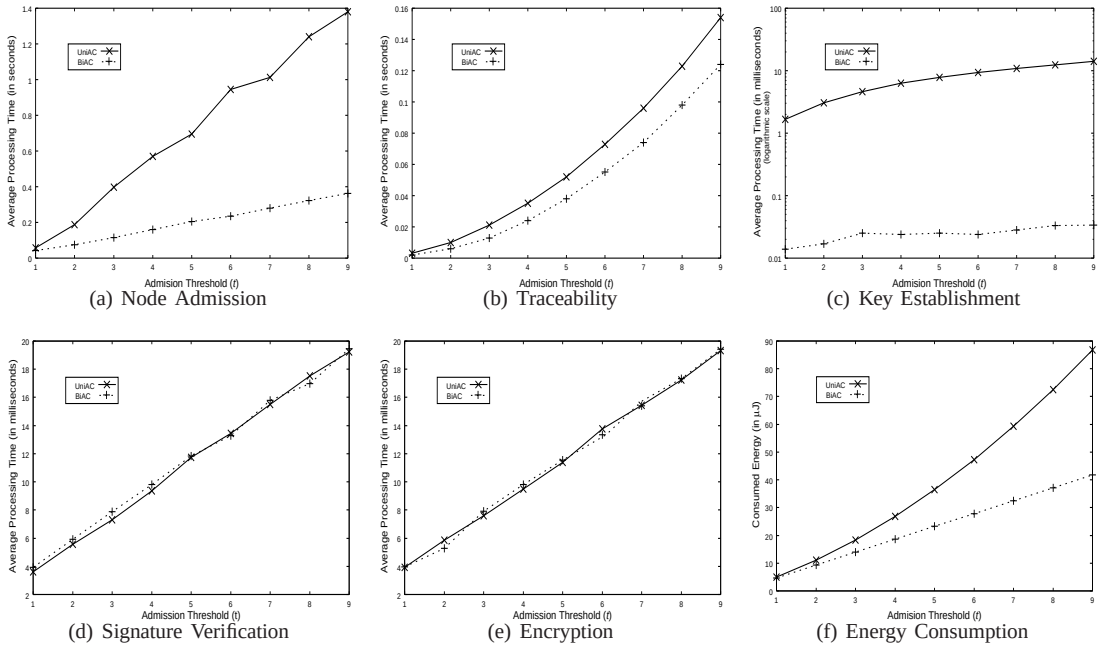
B. Experimental Setup

UniAC and BiAC protocols have been implemented using the popular OpenSSL library [38]. Our imple-

[5]For example, for $n=100$ and $t=10$ (10% of group size), *BiAC-Enc* would require <70 squarings, <70 modular multiplications and only 2 modular exponentiations. The decryption would require only 1 exponentiation. In contrast, IBE encryption requires 1 map-to-point operation, 2 scalar point multiplications and 1 bilinear mapping. IBE decryption costs 1 bilinear mapping. It is well-known that for appropriate security parameters, IBE computations are extremely costly (e.g., a single bilinear mapping takes around 80ms, scalar point multiplication – around 30ms, while a modular exponentiation takes only a few milliseconds. Refer to, e.g., [45] for details regarding these cost comparisons.

[6]Costs related to the signature scheme required for protecting each protocol message are not taken account, since these vary with the specific signature scheme.

[7]We assume that the identity and the public key are $\log q$ bits long and $\log p$ bits long, respectively.



(a) Node Admission (b) Traceability (c) Key Establishment

(d) Signature Verification (e) Encryption (f) Energy Consumption

Fig. 5. Experiment Results. BiAC performs much better than UniAC in node admission, pairwise key establishment, and energy consumption experiments and shows similar performance in the other experiments.

mentation consists of approximately $20,000$ lines of C code running on Linux 2.4. The source is publicly available at [39]. We now describe the experimental setup used for performance measurements. Our experiments were conducted in a *real* wireless MANET environment and included measuring energy costs for each scheme with power measuring system described below.

1) Wireless Mobile Ad Hoc Networks: We use five laptop computers for our wireless experimental setup: four with Pentium-3 800 MHz CPU and 256 MB RAM and one with Mobile Pentium 1.8 GHz CPU and 512 MB RAM. Each laptop is configured with $802.11b$ in ad-hoc mode and runs the Optimized Link State Routing Protocol (OLSR) [37]. Each machine runs Linux version 2.4

2) Power Measurement Systems: To measure battery power consumption, we configured the following equipment. The test machine was an iPAQ (model H5555) running Linux (Familiar-0.7.2). The CPU on the iPAQ is a 400 MHz Intel XScale with 48MB of flash memory and 128MB of SDRAM. To obtain accurate power measurements we removed the battery from the iPAQ during the experiment and placed a resistor in series with a power supply. We used a National Instruments PCI DAQ (Data AcQuisition) board to sample the voltage drops across the resistor to calculate the current at $1,000$ samples per second.

C. Test Methodology

1) Parameter Selection: To perform fair comparisons, we consider the following parameters. The bit-sizes of q and p were set to 160 and 1024, respectively. Measurements were performed with different threshold values t, ranging between 1 to 9. We used 1024-bit RSA signature algorithm with the fixed public exponent $65,537$ $(= 2^{16} + 1)$ for protocol message authentication. All experiments were repeated $1,000$ times for each measurement in order to get accurate average results.

2) Test Cases: We measured separately the costs of admission, traceability, pairwise key establishment, signing, encryption, and energy consumption.

- *Admission:* four laptops with the same computing power were used as current member nodes and the higher-end laptop was used as the joining node. In this experiment, each node (except the joining node) was emulated by a daemon and each machine was running up to three daemons. We then measured total processing time between sending of `JOIN_REQ` by the prospective node and receiving (plus verification) of acquired secret shares. The measurement thus include the

average computation time of the basic operations (e.g., modular multiplications and exponentiations) as well as communication costs, such as packet en/decoding time and network delay.
- *Traceability:* we measured the computation time for tracing partial shares received during the admission protocol. We measured this cost using pre-computed values, to the extent possible.
- *Pairwise Key Establishment:* we measured the time to compute a pairwise key on the higher-end laptop. Note that no communication is involved in this measurement.
- *Signature Verification:* we measured the time for verifying a signature only, since the same method for signature generation has been applied to both UniAC and BiAC.
- *Encryption:* we measured the time to encrypt sample data. Decryption costs were not compared as they are the same for UniACand BiAC.
- *Energy Consumption:*we measured power consumption in terms of communication bandwidth in each admission protocol: we transmitted bulk data (e.g., 100 MB) from a single iPAQ PDA, measured power consumed for transmission, and then computed the average power consumption per bit. After that, we calculated power consumption of each admission protocol by multiplying this measurement result by the bit length of the transmitted data.

D. Experimental Results

- *Admission:* as shown in Figure 5(a), the admission cost in BiAC is much lower than that in UniAC. The difference is even higher for higher threshold values, since BiAC is not only computationally cheaper, but it also requires less communication.
- *Traceability:* Figure 5(b) shows the traceability costs for the two approaches. Even in the worst case, BiAC is as good as UniAC for performing the (very infrequent) operation of tracing malicious nodes.
- *Pairwise Key Establishment:* Figure 5(c) shows that BiAC is much more efficient than UniAC. The differences range from 115 ($t = 1$) to 412 ($t = 9$). In other words, BiAC is $115 - 412$ times faster than UniAC for establishing a shared secret. This result was expected since pairwise key establishment in BiAC requires only $O(t)$ multiplications for a 160-bit modulus. In contrast, UniAC requires $O(t)$ exponentiations with a modulus size of 1024 as well as $O(t)$ multiplications with a 160-bit modulus.
- *Signature Verification:* Figure 5(d) shows that BiAC is as complicated as UniAC in verifying a signature and the cost is proportional to a threshold due to special construction of public key using the witnesses.
- *Encryption:* Figure 5(e) shows that BiAC and UniAC exhibit the same encryption costs.
- *Energy Consumption:* Figure 5(f) clearly illustrates that BiAC is much more energy-efficient than UniAC.

X. Conclusions and Future Work

This paper considered node admission in temporary MANETs and presented BiAC – an efficient and fully non-interactive admission techniques based on bi-variate polynomial secret sharing. We also showed how to obtain efficient public key encryption and signatures as well as establish shared secret keys by treating nodes' secret shares as private keys. We demonstrated – via analytical and experimental evaluation – that our technique compares very favorably to prior results. As part of future work, we plan to explore techniques for improving decentralized group initialization. The currently used JSS protocol [19] is inefficient in terms of communication and requires a reliable broadcast channel. We also intend to address the problem of distributed membership revocation, e.g., to expel malicious nodes from the group.

Appendix I
Proof of Theorem 1

Proof: We prove the following claim: if there exists a polynomial time algorithm $\mathcal{A}$, which on inputs the secret keys of $t-1$ corrupted users, is able to create an existential forgery in CMA model corresponding to an uncorrupted user, then there exists a polynomial time algorithm $\mathcal{B}$, which can break the DL assumption in ROM.

We construct an algorithm $\mathcal{B}$, which runs on input of a DL instance $y = g^x \pmod p$, and would translate the adversarial algorithm $\mathcal{A}$ into outputting x. We first assume that the adversary $\mathcal{A}$ corrupts $t-1$ nodes denoted by $P_1, P_2, \cdots, P_{t-1}$, w.l.o.g.

Note that in our multiple user scenario, the adversary $\mathcal{A}$ can request the signature oracle to sign chosen messages corresponding to any honest node. In other words, when $\mathcal{A}$ sends (m, id_i) to the signature oracle, the oracle responds with a signature on message m signed with x_i.

$\mathcal{B}$ picks $x_1, x_2, \cdots, x_{t-1}$ values corresponding to the secret keys of corrupted users, uniformly at random from $\mathbb{Z}_q$. It then sets $x_i = F(id_i)$, and employs appropriate Lagrange interpolation coefficients in the exponent to compute the public witnesses $g^{A_1}, \cdots, g^{A_{t-1}} \pmod p$, where $F(z) = x + A_1 z + \cdots + A_{t-1} z^{t-1} \pmod q$. Since, $x = \sum_{k=1}^{t-1} x_k \lambda_k(0) + x_i \lambda_i(0) \pmod q$, $\mathcal{B}$ can compute the public key y_i, corresponding to an honest node P_i ($i \geq t$) as

$$y_i = \left\{ \frac{y}{g^{\sum_{k=1}^{t-1} x_k \lambda_k(0)}} \right\}^{1/\lambda_i(0)} \pmod p \quad (6)$$

$\mathcal{B}$ now runs $\mathcal{A}$ on inputs $x_1, x_2, \cdots, x_{t-1}$ and simulates the signature oracle on $\mathcal{A}$'s query (m, id_i), by picking s and c at random in $\mathbb{Z}_q$, computing $r = g^s {y_i}^{-c} \pmod p$ and setting $H(m, r) = c$. $\mathcal{A}$ then outputs a forgery (C, S) on some message M corresponding to user P_i. Note that because H is a random function, except for negligible probability, $\mathcal{A}$ must have asked to H a query (M, R) where $R = g^S {y_i}^{-C} \pmod p$, because otherwise it could not have guessed the value of $C = H(M, R)$. $\mathcal{B}$ then reruns $\mathcal{A}$ by giving the same answers to queries to H until the query (M, R), which it now answers with new randomness C'. If $\mathcal{A}$ outputs the forgery on the same message M, but this time for a different user P_j ($i \neq j$) then, except for negligible probability, it produces S' s.t. $R = g^{S'} {y_j}^{-C'} \pmod p$. $\mathcal{B}$ can now (using Equation (6)) compute $x = \{S - S' + \frac{C}{\lambda_i(0)} \sum_{k=1}^{t-1} x_k \lambda_k(0) - \frac{C'}{\lambda_j'(0)} \sum_{k=1}^{t-1} x_k \lambda_k'(0)\} \; / \; \{\frac{C}{\lambda_i(0)} - \frac{C'}{\lambda_j'(0)}\} \pmod q$.

As in the security proof of Schnorr's Signatures [40], the probability of success of $\mathcal{B}$ would be $\epsilon^2/4q$, where ϵ represents the success probability of $\mathcal{A}$ and q is the total number of queries to $H()$. ■

APPENDIX II
PROOF OF THEOREM 2

Proof: As usual, the proof goes by contradiction, i.e., we proof that if there exists a polynomial time algorithm $\mathcal{A}$, which on inputs the secret keys of $t-1$ corrupted users, is able to break the multi-user indistinguishability notion, then there exists a polynomial time algorithm $\mathcal{B}$, which can break the CDH assumption in ROM.

We construct an algorithm $\mathcal{B}$, which running on input of a CDH instance $U = g^u, V = g^v$, translates the algorithm $\mathcal{A}$ into outputting g^{uv}. As usual, we first assume that the adversary $\mathcal{A}$ corrupts $t-1$ nodes denoted by $P_1, P_2, \cdots, P_{t-1}$, w.l.o.g.

As in the security proof of *BiAC-Sig*, $\mathcal{B}$ picks $x_1, x_2, \cdots, x_{t-1}$ values corresponding to the secret keys of corrupted users, uniformly at random from $\mathbb{Z}_q$. It then sets $x_i = F(id_i)$, and employs appropriate Lagrange interpolation coefficients in the exponent to compute the public witnesses $g^{A_1}, \cdots, g^{A_{t-1}} \pmod p$, where $F(z) = u + A_1 z + \cdots + A_{t-1} z^{t-1} \pmod q$. Since, $u = \sum_{k=1}^{t-1} x_k \lambda_k(0) + x_i \lambda_i(0) \pmod q$, $\mathcal{B}$ can compute the public key y_i, corresponding to an honest node P_i ($i \geq t$) using Equation (6).

To help the reader understand the construction of our translator algorithm $\mathcal{B}$, we first recall the how the translator works in the security proof (under CDH and ROM) of single-user hashed ElGamal. The translator works as follows: on input of a CDH instance $(U = g^u, V = g^v)$, it first runs the adversary on input g^u. The adversary outputs two messages m_0, m_1. The translator picks one message m_b ($b = 0$ or 1) at random, and sends the encryption (c_1, c_2) to the adversary, where $c_1 = Vg^r \pmod p$ and $c_2 = R$ (r is a random value in $\mathbb{Z}_q$ and R is a random pad of same length as the message). In the random oracle model, the only way the adversary can distinguish this encryption is by querying the random oracle on value $O = c_1^u = U^{r+v}$, which will be recorded by the translator, and used to compute $g^{uv} = OU^{-r}$. If there are a total of q queries being made to the oracle, this means that the probability of success of translator would be $1/q$ times the probability of success of the adversary.

Now, we are ready to describe the translation based on our multi-user setting: $\mathcal{B}$ runs $\mathcal{A}$ on inputs the secret keys $x_1, \cdots, x_{t-1}$ corresponding to the corrupted users, and the public keys $y_t, \cdots, y_n$ of all honest ones. $\mathcal{A}$ outputs two vectors of $(n-t+1)$ messages $M_0 = \{m_{0i}\}$ and $M_1 = \{m_{1i}\}$, where $i = t, \cdots, n$, to be challenged upon. $\mathcal{B}$ then picks M_b (b is 0 or 1) and sends to $\mathcal{A}$ the vector $\{(Vg^{r_i}, R_i)\}$, where r_i is a random value in $\mathbb{Z}_q$, and R_i is a random pad equally long as the message m_{bi}, for $i = t, \cdots, n$. The only possibility for $\mathcal{A}$ to win this game, is by querying the random oracle on at least one of the value $O = (Vg^{r_j})^{x_j}$, for some $j \in \{t, \cdots, n\}$. $\mathcal{B}$ records this value, and assuming that it corresponds to P_j, it computes g^{uv} as follows: $u = \sum_{k=1}^{t-1} x_k \lambda_k'(0) + x_j \lambda_j'(0) \pmod q$. This implies that $g^{uv} = g^{v \sum_{k=1}^{t-1} x_k \lambda_k'(0)} g^{v x_j \lambda_j'(0)} \pmod q$ and $g^{uv} = V^{\sum_{k=1}^{t-1} x_k \lambda_k'(0)} V^{x_j \lambda_j'(0)} \pmod p$. Since $O = (Vg^{r_j})^{x_j}$, this means $V^{x_j} = O {y_j}^{-r_j}$, and therefore, $g^{uv} = V^{\sum_{k=1}^{t-1} x_k \lambda_k'(0)} O {y_j}^{-r_j \lambda_j'(0)} \pmod p$.

Given that there are a total of q queries to the random oracle, the probability of success of B would

be probability of success of $\mathcal{A}$ times $1/q(n-t+1)$, as only one query will yield correct g^{uv} value and each query might correspond to one j value in $\{t, n\}$. ■

Remark: Extension to Chosen Ciphertext Security. The hybrid encryption techniques for extending standard hashed ElGamal to chosen ciphertext security (refer to [5], [17]) can be used to achieve chosen ciphertext security for the *BiAC-Enc* scheme.

REFERENCES

[1] K. Barr and K. Asanovic, "Energy Aware Lossless Data Compression," in *ACM MobiSys'03*, pp. 231–244, 2003.

[2] O. Baudron, D. Pointcheval, and J. Stern, "Extended Notions of Security for Multicast Public Key Cryptosystems," in *ICALP'00*, pp. 499–511, 2000.

[3] M. Bellare and P. Rogaway, "Random Oracles are Practical: A Paradigm for Designing Efficient Protocols," in *ACM CCS'93*, pp. 62–73, 1993.

[4] M. Bellare, A. Boldyreva, and S. Micali, "Public-key encryption in a multi-user setting: Security proofs and improvements." in *EUROCRYPT'00*, pp. 259–274, 2000.

[5] M. Bellare, A. Boldyreva, and A. Palacio, "An Uninstantiable Random-Oracle-Model Scheme for a Hybrid Encryption Problem," in *EUROCRYPT'04*, pp. 171–188, 2004.

[6] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation," in *ACM STOC'88*, pp. 1–10, 1988.

[7] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences," in *CRYPTO'92*, pp. 471–48, 1992.

[8] A. Boldyreva, "Efficient Threshold Signatures, Multisignatures and Blind Signatures based on the Gap-Diffie-Hellman-Group Signature Scheme," in *PKC'03*, pp. 31–46, 2003.

[9] D. Boneh and M. K. Franklin, "Identity-Based Encryption from the Weil Pairing," in *CRYPTO'01*, pp. 213–229, 2001.

[10] G. Bracha, "An Asynchronous $\lfloor (n-1)/3 \rfloor$-resilient Consensus Protocol," in *ACM PODC'84*, pp. 154–162, 1984.

[11] C. Castelluccia, N. Saxena, and J. H. Yi, "Self-Configurable Key Pre-distribution in Mobile Ad Hoc Networks," in *IFIP Networking'05*, pp. 1083–1095, 2005.

[12] J. C. Cha and J. H. Cheon, "An ID-based Signature from Gap-Diffie-Hellman Groups," in *PKC'03*, pp. 18–30, 2003.

[13] B. Dahill, B. Levine, E. Royer, and C. Shields, "A secure routing protocol for ad hoc networks," University of Massachusetts, Technical Report UM-CS-2001-037, 2001.

[14] Y. Desmedt and Y. Frankel, "Threshold Cryptosystems," in *CRYPTO'89*, pp. 307–315, 1989.

[15] ElGamal, "A Public-Key Cryptosystem and A Signature Scheme based on Discrete Logarithms," in *IEEE ToIT*, pp. 469–472, 1985.

[16] P. Feldman, "A Practical Scheme for Non-interactive Verifiable Secret Sharing," in *IEEE FOCS'87*, pp. 427–437, 1987.

[17] E. Fujisaki and T. Okamoto, "Secure Integration of Asymmetric and Symmetric Encryption Schemes," in *CRYPTO'99*, pp. 537–554, 1999.

[18] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Robust Threshold DSS Signatures," in *CRYPTO'96*, pp. 354–371, 1996.

[19] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure Distributed Key Generation for Discrete-Log Based Cryptosystems," in *EUROCRYPT'99*, pp. 295–310, 1999.

[20] S. Goldwasser and S. Micali, "Probabilistic Encryption," *Elsevier JCSS*, vol. 28, pp. 270–299, 1989.

[21] S. Goldwasser, S. Micali, and R. L. Rivest, "A Paradoxical Solution to the Signature Problem," in *IEEE FOCS'84*, pp. 441–448, 1984.

[22] S. Goldwasser, S. Micali, and R. L. Rivest, "A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 281–308, 1988.

[23] M. T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun, "Loud and Clear: Human-Verifiable Authentication Based on Audio," in *ICDCS'06*, 2006.

[24] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive Secret Sharing, Or How To Cope With Perpetual Leakage," in *CRYPTO'95*, pp. 339–352, 1995.

[25] R. Hills, "Sensing for Danger," Science Technology Report, July/August 2001, http://www.llnl.gov/str/JulAug01/Hills.html.

[26] Y.-C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," in *IEEE WMCSA'02*, pp. 3–13, 2002.

[27] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," in *ACM MobiCom'02*, pp. 12–23, 2002.

[28] S. Jarecki, N. Saxena, and J. H. Yi, "An Attack on the Proactive RSA Signature Scheme in the URSA Ad Hoc Network Access Control Protocol," in *ACM SASN'04*, pp. 1–9, 2004.

[29] J. Kong, H. Luo, K. Xu, D. L. Gu, M. Gerla, and S. Lu, "Adaptive Security for Multi-level Ad-hoc Networks," in *Wiley WCMC*, vol. 2, no. 5, pp. 533–547, 2002.

[30] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, "Providing Robust and Ubiquitous Security Support for MANET," in *IEEE ICNP'01*, pp. 251–260, 2001.

[31] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," in *ACM CCS'03*, pp. 52–61, 2003.

[32] H. Luo, J. Kong, P. Zerfos, S. Lu, and L. Zhang, "URSA: Ubiquitous and Robust Access Control for Mobile Ad Hoc Networks," in *IEEE/ACM ToN*, vol. 12, no. 6, pp. 1049–1063, 2004.

[33] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-securing Ad Hoc Wireless Networks," in *IEEE ISCC'02*, pp. 567–574, 2002.

[34] J. M. McCune, A. Perrig, and M. K. Reiter, "Seeing-Is-Believing: Using Camera Phones for Human-Verifiable Authentication," in *IEEE SP'05*, pp. 110–124, 2005.

[35] M. Naor, B. Pinkas, and O. Reingold, "Distibuted Pseudo-Random Functions and KDCs," in *EUROCRYPT'99*, pp. 327–346, 1999.

[36] M. Narasimha, G. Tsudik, and J. H. Yi, "On the Utility of Distributed Cryptography in P2P and MANETs: The Case of Membership Control," in *IEEE ICNP'03*, pp. 336–345, 2003.

[37] OLSR Protocol, http://menetou.inria.fr/olsr.

[38] OpenSSL Project, http://www.openssl.org.

[39] Peer Group Admission Control Project, http://sconce.ics.uci.edu/gac.

[40] D. Pointcheval and J. Stern, "Security Proofs for Signature Schemes," in *EUROCRYPT'96*, pp. 387–398, 1996.

[41] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C : The Art of Scientific Computing*, Cambridge University Press, ISBN 0-521-43108-5, 1992.

[42] N. Saxena, "Public Key Cryptography Sans Certificates in Ad Hoc Networks," in *ACNS'06*, pp. 375–389.

[43] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan, "Secure Device Pairing based on a Visual Channel (Short Paper)," in *IEEE SP'06*, pp. 306–313, 2006.

[44] N. Saxena, G. Tsudik, and J. H. Yi, "Admission Control in Peer-to-Peer: Design and Performance Evaluation," in *ACM SASN'03*, pp. 104–114, 2003.

[45] N. Saxena, G. Tsudik, and J. H. Yi, "Identity-based Access Control for Ad-Hoc Groups," in *ICISC'04*, pp. 362–379, 2004.

[46] N. Saxena, G. Tsudik, and J. H. Yi, "Efficient Node Admission for Short-lived Mobile Ad Hoc Networks," in *IEEE ICNP'05*, pp. 269–278, 2005.

[47] N. Saxena, G. Tsudik, and J. H. Yi, "Threshold Cryptography in P2P and MANETs: the Case of Access Control," in *Computer Networks*, vol. 51, pp. 3632–3649, 2007.

[48] C. P. Schnorr, "Efficient Signature Generation by Smart Cards," in *Journal of Cryptology*, vol. 4, no. 3, pp. 161–174, 1991.

[49] A. Shamir, "How to Share a Secret," in *CACM*, vol. 22, no. 11, pp. 612–613, 1979.

[50] M. Steiner, G. Tsudik, and M. Waidner, "CLIQUES: A New Approach to Group Key Agreement," in *IEEE ICDCS'98*, pp. 380–387, 1998.

[51] M. Steiner, G. Tsudik, and M. Waidner, "Key Agreement in Dynamic Peer Groups," in *IEEE TPDS*, vol. 11, no. 8, pp. 769–780, 2000.

[52] D. R. Stinson and R. Strobl, "Provably Secure Distributed Schnorr Signatures and a (t, n) Threshold Scheme for Implicit Certificates," in *ACISP'01*, pp. 417–434, 2001.

[53] E. Uzun, K. Karvonen, and N. Asokan, "Usability Analysis of Secure Pairing Methods," in *USEC'07*, pp. 15–16, 2007.
[54] P. van Oorschot, "Extending Cryptographic Logics of Belief to Key Agreement Protocols," in *ACM CCS'93*, pp. 232–243, 1993.
[55] L. Zhou and Z. J. Haas, "Securing Ad Hoc Networks," in *IEEE Network Magazine*, vol. 13, no. 6, pp. 24–30, 1999.

Nitesh Saxena is an Assistant Professor in the Department of Computer and Information Science at Polytechnic University. He works in the areas of computer and network security, and applied cryptography. Nitesh obtained his Ph.D in Information and Computer Science from UC Irvine. He holds an M.S. in Computer Science from UC Santa Barbara, and a Bachelor's degree in Mathematics and Computing from the Indian Institute of Technology, Kharagpur, India. Nitesh's Ph.D. dissertation on "Decentralized Security Services" has been nominated for the ACM Dissertation Award 2006. Nitesh received the "best student paper" award at ACNS 2006.

Gene Tsudik is a Professor of Computer Science at the University of California, Irvine (UCI) and the Director of Secure Computing and Networking Center (SCONCE). He has been conducting research in internetworking, network security and applied cryptography since 1987. He obtained a PhD in Computer Science from USC in 1991; his dissertation focused on access control in internetworks. Before coming to UCI in 2000, he was a Project Leader at IBM Research, Zurich Laboratory (1991-1996) and USC Information Science Institute (1996-2000). Between 2002 and 2007, he served as Associate Dean of Research and Graduate Studies in the School of Information and Computer Sciences at UCI. In 2007, Gene Tsudik was a Fulbright Senior Lecturer at the University of Rome (La Sapienza). Over the years, his research interests included: routing, firewalls, authentication, mobile/wireless network security, secure e-commerce, anonymity, secure group communication, digital signatures, key management, secure ad hoc and sensor network routing, as well as database privacy and secure storage.

Jeong Hyun Yi is a Principal Researcher at Samsung Advanced Institute of Technology (SAIT), Korea. He obtained a Ph.D. in Information and Computer Science from University of California, Irvine (UCI) in 2005. He received an M.S. and a B.S. in Computer Science at Soongsil University, Korea in 1995 and 1993, respectively. He was a member of research staff at Electronics and Telecommunication Research Institute (ETRI), Korea, from 1995 to 2001 and a guest researcher at National Institute of Standards and Technology (NIST), USA, from 2000 to 2001. His research interests include: network security, ubiquitous computing, RFID, wireless sensor networks, mobile privacy, PKI, and applied cryptography. Some of his notable research contributions include Certificate Management Protocol (CMP) for Korean PKI Standards and integration of Korea PKI and US Federal PKI.