A Survey of Key Management in Ad Hoc Networks

A.M. Hegland, E. Winjum, S.F. Mjølsnes, C. Rong, Ø. Kure, and P. Spilling

***Abstract*—The wireless and dynamic nature of mobile ad hoc networks (MANETs) leaves them more vulnerable to security attacks than their wired counterparts. The nodes act both as routers and communication end-points. This makes the network layer more prone to security attacks. A main challenge is to judge whether a routing message originates from a trustworthy node or not. The solution so far is cryptographically signed messages. The general assumption is that nodes are in possession of a valid secret key can be trusted. Consequently, a secure and efficient key management scheme is crucial.**

Keys are also required for protection of application data. However, the focus here is on network layer management information. Whereas key management schemes for the upper layers can assume an already running network service, schemes for the protection of the network layer cannot. Keys are a prerequisite to bootstrap a protected network service.

This paper surveys the state of the art within key management for ad hoc networks, and analyzes their applicability for network layer security. The analysis puts some emphasis on their applicability in scenarios such as emergency and rescue operations as this work was initiated by a study of security in MANETs for emergency and rescue operations.

***Index Terms*—Ad hoc networks, key management, security, network layer, emergency and rescue operations.**

I. Introduction

MOBILE ad hoc networks have wireless links and work independently of fixed infrastructure. They are self-organizing and self-configuring. The wireless nodes operate both as communication end-points as well as routers, enabling multi-hop wireless communication. The wireless devices imply limited power resources and bandwidth. Network topology may change rapidly due to mobility, interference, physical obstacles on the path, etc. Application areas range from conference hall networks to ad hoc networks for emergency and rescue operations and military tactical use.

The wireless and dynamic nature of ad hoc networks leaves them more vulnerable to security attacks than their wired counterparts. Passive eavesdropping as well as active message insertions, denial of service and battery-exhaustion attacks are inherently easy. Security attacks can be launched towards any layer of the protocol stack. Defense mechanisms for the lowest layers call for physical tamper protection and transmission security measures as spread spectrum techniques, frequency hopping, and interleaving. Cryptographic techniques are essential for the protection of the higher layers.

In wired networks the routers are part of an established and (at least to some extent) controllable infrastructure. The same is not true in ad hoc networks where the nodes act both as routers and communication end-points. This makes the network layer more prone to security attacks. There is no guarantee that malicious nodes do not mingle and interfere. Examples of possible attacks are misdirection and insertion of bogus routing information, black holes (nodes attracting traffic by maliciously advertising shortest path to other nodes), and wormholes (adversary nodes colluding by tunneling packets from one part of the network to another).

A primary challenge is to decide which routing information can be trusted. A number of schemes relying on cryptographically signed routing messages have been designed – most without detailing key management further. Nevertheless, the possession of cryptographic keys serves as proof of trustworthiness. Consequently, a proper key management service is required. This is to

This work was sponsored by Kongsberg Defence Communications and the Norwegian Research Council, grant no. 156713/220.

ensure that nodes which are legitimate members of the network – and only those – are equipped with the necessary keys whenever needed. Whereas key management services are needed for application layer security as well as for protection of the network layer, this paper focuses on the more challenging of the two, namely providing keys for the network layer. Key management schemes for the application layer can assume an already running network service. Schemes for the network layer routing information cannot. Keys are a prerequisite to bootstrap a protected network service.

This survey was motivated by an investigation of security in ad hoc networks for emergency and rescue operations. Most of the discussions have relevance for mobile ad hoc networks (MANETs) in general. However, emergency and rescue operations have some additional requirements. Where appropriate, concerns regarding the applicability of the various key management schemes in settings with the characteristics of emergency and rescue operations are highlighted.

Emergency and rescue operations imply MANETs with nodes that have gone through a preparation phase prior to MANET initialization. That is, pre-distribution of keys and other parameters is possible. MANETs for emergency and rescue operations present stronger requirements than most commercial applications. Time is scarce. When the rescue team arrives on the scene of an accident, communication should be established immediately and maintained with as little human interaction as possible. Availability is a number one requirement.

The network resources should be reserved for the members of the emergency and rescue team, and not used to convey arbitrary data for others. It should be possible to distinguish legitimate nodes from untrustworthy ones and build a reliable route through trusted nodes only.

The structure of emergency and rescue operations has implications for key management:

--**Single administrative domain involved (SAD)**

SAD operations refer to operations where all involved parties belong to the same regime or share a common, predefined point of trust. Local, regional or national rescue operations including only predefined actors are in this category. This setting enables pre-configuring of security credentials.

--**Multiple administrative domains involved (MAD)**

MAD operations represent operations involving ad hoc partners. That is parties that have had no prior contact and belong to different organizational/security domains. This means cases where overall pre-configuring of security parameters is not possible. Examples include cross-border operations and operations involving industrial companies.

Standards: None of the emerging MANET Internet-drafts and RFCs have so far encompassed key management. Of other standards, the IEEE 802.11i [1] security amendment for IEEE 802.11wireless local area networks assumes keys are pre-shared or established with the aid of fixed infrastructure. In case of truly ad hoc communication, pre-shared symmetric keys are the only option. The aim of IEEE 802.11i is protection of *payload* (data frames) on layer 2. IEEE has in 2005 begun work on 802.11w that will cover security on *management* frames. Other standards for wireless communication include the ZigBee[2]/IEEE 802.15.4[3] and the Bluetooth[4] specifications for personal area networks. The preconditions of these standards are infrastructure-based networks and do not apply to MANETs. ZigBee specifies key management for the security elements of IEEE 802.15.4. ZigBee assumes the initial keys are pre-distributed, installed out-of-band, or received in the clear over the air from a trust center. Keys in Bluetooth are derived with the aid of PIN codes. A common PIN code is entered out-of-band in pairs of nodes that wish to communicate. Standards for key management in ad hoc networks lack.

The contribution of this paper is a survey of proposed key management schemes for ad hoc networks, and an analysis of their applicability for MANET network layer security –with some emphasis on their applicability in MANETs for emergency and rescue operations.

The rest of the paper is organized as follows. Desirable features for key management schemes in MANETs are described in section II. Section III classifies and evaluates proposed key management schemes. Conclusions are found in section IV.

II. Desirable features of MANET key management schemes

The evaluation parameters reflect the intention of bootstrapping a protected MANET network service. Evaluations of key management protocols often focus on computational complexity. However, with the wireless media, bandwidth is regarded as a more constrained resource than computational power. A key management service for protection of the network layer should not demand an already running network service or overall connectivity.

--Applicability: The various key management schemes focus on different targets. The aim may range from group key establishment to availability of central management entities. Their applicability for SAD and MAD operations depends on the fundamental assumptions as to network origin (planned or truly ad hoc), network size, node mobility, geographic range and the required level of human involvement.

--Security: *Authentication* and *intrusion tolerance* is a primary concern to ensure no unauthorized node receives key material that can later be used to prove status as a legitimate member of the network. Nobody should provide private keys or issue certificates for others unless the others have been authenticated. Intrusion tolerance means system security should not succumb to a single, or a few, compromised nodes. Other central security issues are *trust management* and *vulnerability.* Trust relations may change during network lifetime. The system should enable exclusion of compromised nodes. In order to judge the security of a key management scheme, possible vulnerabilities should be pinpointed. Proper key lengths and cryptographic algorithms of adequate strength are assumed.

--**Robustness:** The key management system should survive despite denial of service attacks and unavailable nodes. The key management operations should be able to be completed despite faulty nodes and nodes exhibiting Byzantine behavior, i.e. nodes that deliberately deviate from the protocol. Necessary key management operations caused by dynamic group changes should execute in a timely manner. Key management operations should not require network-wide and strict synchronization.

--Scalability: Key management operations should finish in a timely manner despite a varying number of nodes and node densities. The fraction of the available bandwidth occupied by network management traffic should be kept as low as possible. Any increase in management traffic reduces available bandwidth for payload data accordingly. Hence, scalability of key management protocols is crucial.

--Simplicity: Simplicity regarding user-friendliness and communication overhead is an additional intuitive and overall critical factor to the success of a key management scheme. We reckon, however, that a system that is secure, robust and scalable implies simplicity. Given that these conditions are fulfilled, we believe simplicity is first and foremost a matter of implementation.

The ideal key management service for ad hoc networks should be simple, formed on the fly, never expose or distribute key material to unauthorized nodes, ensure that system security does not succumb to (a few) compromised nodes, easily allow re-keying/key updates, enable withdrawal of keys when nodes are compromised or keys for other reasons should be revoked, be robust to Byzantine behavior and faulty nodes, scale well enough to handle the expected network sizes and node densities, and efficiently manage network splits and joins.

Signed routing information requires a security relation that allows one-to-many signing and verification. Routing messages are often broadcast, and all receiving nodes should be able to check the validity. Messages such as neighbor detection messages are not forwarded by other nodes. Other routing messages, such as topology information messages in proactive routing protocols and route requests and route replies in reactive routing protocols, are flooded into the entire network. The receiving nodes may not be known to the transmitting node. In addition, bandwidth is limited. Unique signatures for each receiver scale badly. In other words, pairwise keys provide no good option for protection of routing information.

III. Overview of Key Management Schemes

Key management schemes can be classified in several ways. In this paper we have the main categories *contributory* and *distributive*, rather than the more commonly used contributory and centralized. The distributive category is here defined to encompass schemes where each key originates from a single node. The nodes may very well co-operate during key distribution, but any key originates from a single source.

Distributive schemes may be centralized, but can also be distributed. In the latter, each node generates a key and tries to distribute it to others. In contributory schemes, the key is a result of a collaborative effort of more nodes. Some of the contributory schemes studied here rely on a centralized entity, others do not. Altogether, we chose the categories “contributory” and “distributive” as we found this classification best reflected the origin of the keys in the schemes studied here. Our classification is illustrated in Fig. 1.

The distributive category is divided into symmetric and public key schemes. Public key schemes include traditional certificate based and identity based schemes. The symmetric schemes are classified as either MANET schemes or WSN (wireless sensor networks) schemes. WSNs represent a new class of ad hoc networks with more constrained nodes than traditional MANETs.

Contributory schemes are characterized by the lack of a trusted third party responsible for generation and distribution of the cryptographic keys. Instead, all communicating parties co-operate to establish, i.e. “agree” upon, a secret symmetric key. The number of participants ranges from two parties (establishing a pairwise key) to many parties (establishing a group key). Although not necessarily designed with ad hoc networks in mind, intuitively the contributory approach of collaboration and self-organization may seem to fit the nature of ad hoc networks. A number of contributory schemes are therefore reviewed and evaluated in section III A. Only one of these was designed specifically for ad hoc networks.

Distributive schemes involve one or more trusted entities and comprise both public key systems and symmetric systems. Truly ad hoc networks require the trusted entity to be established impromptu during network initialization.

Certificate-based public key schemes require the public keys to be distributed in a way that allows the receiving nodes to verify the authenticity of the key material. The wired network solution is a public key infrastructure (PKI) where a centralized certificate authority (CA) issues certificates binding the public keys to specific users/nodes.

If it is suspected that a node has fallen into the wrong hands, or the node for other reasons should be expelled, the certificate is revoked. Revoked certificates are added to the certificate revocation list (CRL). The CA signature guarantees the authenticity of certificates and CRLs. Under the assumption that a centralized trusted entity is not well suited for ad hoc networks where overall availability cannot be guaranteed all the time, the proposed key management schemes for ad hoc networks involving certificate based PKI, advocate various ways to distribute the CA functionality. The intuitive approach of naive CA replication is not reckoned as good enough as it poses poor intrusion tolerance. With more nodes holding the private CA key, the higher is the risk of getting it compromised.

Identity-based public key schemes [5] represent a new type of public key system. They allow user identities, e.g. e-mail - or IP addresses, to be used as public keys, and make certificates superfluous. A trusted entity is however required in order to generate and distribute the private keys corresponding to the various identities. The trusted entity is also needed for revocation. The trusted entity may sign a list of withdrawn identities. As with traditional public key systems, it has been suggested to spread the trusted entity over more nodes.

Symmetric systems aim at distributing one or more shared secrets through secure channels. Many of the symmetric key management systems for ad hoc networks found in the literature are intended for Wireless Sensor Networks (WSNs). The sensor nodes possess very limited power, memory and computational resources compared to traditional MANET nodes. Symmetric systems may thus be the only option. WSNs normally include a base station. That is, WSNs have a certain amount of infrastructure and are thus not truly ad hoc networks. This survey distinguishes between symmetric schemes for traditional MANETs and WSN schemes. A number of WSN schemes have been included in order to evaluate their applicability in traditional MANETs.

Related surveys of key management schemes: A survey of key distribution mechanisms for wireless sensor networks is found in [6]. Key Management schemes for secure group communication are surveyed in [7]. Reviews of key management protocols for ad hoc networks and sensor networks are also found in [8],[9],[10], and [11].

A. Contributory Schemes

The main implications and limitations of various types of contributory schemes in ad hoc networks are demonstrated by the schemes studied in this section. TABLE I summarizes the features of the different schemes.

1) Diffie-Hellman (D-H)[1] [12]*:* D-H establishes a unique symmetric key between two parties. It relies on the discrete log problem (DL); deciding *S* given *g^S mod p* being a hard problem. D-H is outlined in Fig. 2. The parties agree upon a large prime, *p*, and a generator, *g*. Each party randomly chooses a secret, S_A and S_B, and transmits the public values, $(g^\wedge S_A)\ mod\ p$ and $(g^\wedge S_B)\ mod\ p$, as shown in the figure. Raising the number received from the other party to the power of its own secret, gives a common secret key, $g^\wedge (S_A S_B)\ mod\ p$, shared only by the two.

Like any schemes involving pairwise unique keys, D-H provides intrusion tolerance. A captured node only compromises the keys it shares with its communicating peers. Byzantine and faulty nodes basically only disturb their own key establishment with communicating peers. D-H is vulnerable to man-in-the middle (MIM) attacks. It is left for the nodes to judge who to trust. But as authentication lacks, Alice cannot be sure that she actually communicated with Bob and not Charlie.

The generic D-H scheme is not applicable for protection of routing information in ad hoc networks. It applies to two parties only. Protection of routing messages with pairwise keys necessitates a different signature for each possible recipient, which scales badly. D-H has been included in this survey solely because the majority of the contributory schemes are founded on this scheme. They basically seek to remedy the shortcomings of D-H regarding MIM vulnerability and extendibility to more than two parties.

2) Ingemarsson, Tang and Wong (ING) [13]*:* ING provides a symmetric group key by extending the 2-participant D-H scheme to *n* participants. Fig. 2 shows the principles with 4 nodes. All nodes are arranged into a logical ring. After *n-1* rounds, each node can calculate the secret key. Each round involves an exponentiation from every node, and every node must transmit its share to the next node in the logical ring as shown in the figure.

ING lacks authentication and is vulnerable to MIM attacks. It scales poorly. Communicational complexity grows proportionally to the number of nodes squared. Byzantine behavior or faulty nodes may inhibit successful key establishment. A captured node means the group key is compromised, and necessitates a re-keying. The scheme does not specify how compromised nodes can be detected. The requirement for the nodes to organize into a logical ring during the key agreement procedure makes ING unsuitable for ad hoc networks. The establishment of keys for protection of routing information implies a logical ring of 1-hop neighbors only (all nodes within direct transmission range). With mobile nodes and unstable links it is questionable whether ING will ever complete successfully.

3) Burmester and Desmedt (B-D) [14]*:* B-D seeks to establish a group key. It relies on the DL problem. But, contrary to the other contributory schemes studied here, it is not based on D-H. An outline of B-D with 4 nodes is shown in Fig. 2. B-D completes in three rounds. Every node picks a secret, S_i , and multicasts its *public value*, $Z_i=g^\wedge S_i$, to all other nodes in the group. In round 2, every node calculates and multicasts a new public value. This value is derived by dividing the public value received from the next node with the public value received from the previous node in the logic ring of nodes, and raising the result to the power of its own secret, S_i, as illustrated in the figure. In the third and final round every node calculates the conference key from its secret and the information received from all the other nodes in the previous rounds.

B-D is apparently more efficient than ING as it completes in three rounds. However, each round requires a high number of exponentiations and reliable multicast. Reliable multicasting is difficult in wired networks, and even more challenging in ad hoc networks. Changes in group membership necessitate a re-start of the key-agreement procedure. In an ad hoc network with moving nodes it may thus never be possible to establish a group key by B-D, nor handle later changes in group membership. Group changes will certainly cause delay and disruption. B-D

[1]For simplicity, the key management schemes are given short names. The short names do not necessarily represent generally adopted abbreviations.

also demands an already running routing protocol or only 1-hop neighbors, i.e. the key agreement schemes depend on an already established routing infrastructure – but the infrastructure cannot be established before the keys have been set up. B-D authentication of the public values (not shown in the figure) can be implemented with the aid of pre-distributed public keys. Trust is managed through the certificate issuer. This implies a planned network and the basic key management problem reverts to a public key scheme.

4) Hypercube and Octopus (H&O) [15]*:* H&O reduces the number of rounds and exponentiations of ING from n to d ($n=2^d$) by arranging the nodes in a *hypercube*, i.e. a d-dimensional cube. Fig. 2 illustrates H&O in a network with 4 (2^2) nodes. In step 1, nodes 1 and 2 perform a D-H key agreement. Nodes 3 and 4 do the same. The symmetric keys established in step 1 are used as the secret values in a new D-H key agreement in step 2: node 1 and 4 perform a D-H key agreement and node 2 and 3 do the same and so on. H&O actually consists of two protocols: Hypercube and Octopus. Hypercube (shown in the figure) assumes the number of participants is a power of 2. Octopus extends Hypercube to allow an arbitrary number of nodes.

H&O is vulnerable to MIM attacks as authentication is absent. Byzantine or faulty nodes may preclude successful key agreement. Changes in group membership require re-keying. It is left for the nodes to decide when re-keying is needed. Like B-D and ING, H&O relies on an underlying communication system to provide a consistent node-ordering view to all group members. Besides the difficulty of keeping a consistent node ordering where nodes join and leave dynamically, it implies an already running (unprotected) routing protocol or only 1-hop neighbors. The latter scales badly. Altogether, H&O is unsuitable for network layer security in ad hoc networks.

5) Password authenticated key agreement (A-G) [16]*:* A-G is the only one of the contributory systems studied that has been designed with ad hoc networks in mind. A-G is basically H&O extended with password authentication as indicated in Fig. 2. It assumes all legitimate participants receive a password off-line (written on the conference hall black-board or distributed through another location limited channel). The nodes must prove knowledge of the password during the pairwise D-H key agreements of the H&O protocols as shown in Fig. 2. The figure shows the password authenticated key agreement between two nodes. The password is used to encrypt the public value and an initial challenge in a challenge-response protocol as illustrated in the figure.

A-G doubles the number of messages and increases the computational complexity compared to H&O. It remedies H&O's vulnerability to MIM attacks at the price of scalability.

A-G inherits the deficiencies of H&O regarding dependability of an already established communication infrastructure and node ordering scheme. Hence, it is not suitable for network layer security in mobile ad hoc networks.

6) CLIQUES (CLIQ) [17] [18]*:* CLIQ is outlined in Fig. 2. It extends the generic D-H protocol to support dynamic group operations. CLIQ distinguishes between Initial Key Agreement (IKA) and Auxiliary Key Agreement (AKA). IKA takes place at group formation. AKA handles all subsequent key agreement operations. In both cases, a group controller synchronizing the key agreement procedure is required.

The figure shows the IKA protocol with four nodes. Stage 1 (the *upflow* stage) starts from node 1 which picks a secret exponent, S_1, and unicasts $g^\wedge S_1$ to next node. Node 2 picks a secret exponent S_2, and unicasts to node 3 the values shown in the figure. The procedure is repeated until the final node – the *group controller* is reached. The group controller is now able to calculate the secret group key, i.e. the generator, g raised to the power of the secret exponents of all nodes in the group. In stage 2 (the *downflow* stage) the group controller multicasts the intermediate values required by each of the other nodes to calculate the secret group key, as shown in the figure.

Both AKA (not shown in the figure) and IKA rely on the group controller. The group controller of CLIQ thus represents a single point of failure. Each AKA operation results in a new group key that is independent of all previous keys. Adding a new member with AKA basically extends stage 1 of the IKA protocol with one node. The role of the group controller can be fixed or floating. Allowing any node to take over the role as group controller renders the system vulnerable to malicious nodes. CLIQ omits authentication. The designers have left security properties such as authentication out while focusing on group changes, but argue that authentication could easily be added. Other major drawbacks with CLIQ, as with B-D, are dependency upon reliable multicast and availability of a consistent view of node ordering. With variable connectivity it is questionable whether IKA and AKA would ever complete successfully. With unstable links, highly mobile nodes and rapid splits and joins, instability may result.

7) Other contributory schemes: A large number of key agreement schemes relying on already distributed keys have been proposed. The basic key management problem thus reverts to distribution of the initial keys. Several schemes are also two-party protocols unsuitable for network layer security and are therefore left out of further discussions. Examples include MQV [19] based on traditional public keys, schemes relying on identity-based public keys such as [20] and [21], and the D-H based protocols proposed in [22].

8) Summary of the contributory key management schemes: Although the contributory approach at first glance may seem to fit the self-organizing nature of ad hoc networks, none of the contributory schemes are good candidates for key management in ad hoc networks. D-H, ING and H&O can be skipped due to missing authentication. They are vulnerable to MIM attacks. B-D and CLIQ can be left out – no matter whether the authentication scheme is included or not – as they have an inherent survivability problem with the dependency on reliable multicasting. A-G fails on scalability and robustness due to the dependency upon node ordering, and availability of all nodes during group changes.

B. Distributive Schemes

This section surveys public key and symmetric key management schemes proposed for ad hoc networks. TABLE II and TABLE III summarize the features of the public key schemes and the symmetric schemes, respectively.

1) Public Key Schemes

--***Partially distributed Threshold CA Scheme (Z-H)***[23]: Z-H assumes a PKI system and puts forward a framework to provide an available, intrusion tolerant, and robust CA functionality for ad hoc networks. The private CA key is distributed over a set of *server* nodes through a (*k,n*) *Secret sharing scheme* [24]. The private CA key is shared between *n* nodes in such a way that at least *k* nodes must co-operate in order to reveal the key. (Finding the private CA key *S* is comparable to finding *f(0)* given a polynomial *f(x)* of degree *k-1* and knowing *k* values, e.g. *f(1), f(2) … f(k)*.)

When queried, each server generates a partial signature of the certificate using its private key share in a *threshold signature scheme* [25]. A server acting as combiner collects the partial signatures and produces a valid signed certificate.

Z-H advises *share refreshing* to counter *mobile adversaries*, i.e. adversaries that temporarily compromise one server and then attack the next. *Proactive secret sharing schemes* [26] allow the shareholders to periodically refresh their shares through collaboration. An adversary must thus compromise more than *t* shares *between* refreshes in order to compromise the system. The original secret does not change, only the shares held by the servers. (Bear in mind the homomorphic property: If $(s_1,s_2...,s_n)$ is a *(k,n)* sharing of *S* and $(a_1,a_2...,a_n)$ is a *(k,n)* sharing of *A*, then $(s_1+a_1, s_2+a_2,...,s_n+a_n)$ is a secret sharing of *S*+*A* [27]. Choosing A=0 gives a new sharing of S). The scheme is made robust to missing and erroneous shares through *verifiable secret sharing* [28]: Extra public information testifies to the correctness of each share without disclosing the share.

Although not clearly stated, the system relies on a central trusted *dealer* to bootstrap the key management service, and decide which nodes shall act as servers. Z-H assumes an underlying (unsecured) routing protocol.

According to [23] nodes cannot get the current public keys of other nodes or establish secure communication with others if the CA service is unavailable. However, every node should hold a copy of its own certificate. For network layer security, it would be more efficient to receive the certificates directly from the communicating peers (or other nodes in the neighborhood). If the certificate is needed to verify a signature on routing information, the node in question must certainly be available; otherwise there would be no requirement to verify its routing message. Thus, the need for on-line CA access is limited. Every node must contact the CA to get its initial certificates (and receive the public key of the CA). The same is true if the node for some reason has lost its private key or has had its certificate revoked. However, to get a new certificate, the node should be authenticated by the CA service – which necessitates some sort of physical contact between the node and the CA service. Certificate *updates* call for CA service. For scenarios like emergency and rescue operations, it would be better to make sure certificates are renewed in the preparation phase and not during network operation.

The CA service is needed for revocation and distribution of CRLs. Z-H postulates that public keys of nodes that are no longer trusted, or have left the network, should be revoked. In an ad hoc network it can be hard to decide when a node has actually left the network. Revoking keys due to temporal missing connectivity would not be wise. More important is revocation of keys belonging to captured nodes. The frequency of such revocations in networks for emergency and rescue operations will expectedly be low.

Periodical share refreshing implies some form of synchronization. Synchronization is bandwidth consuming and difficult in ad hoc networks. Management traffic between server nodes and certificate exchanges also consumes much bandwidth, and makes Z-H scale badly. A single CA or hierarchy of CAs is likely to prove better than the Z-H approach. SAD operations allow pre-distributed certificates. MAD operations call for on-scene cross-certification of the root CAs of the merging domains. Efficient spreading of the cross-certificates in the respective domains is a problem for further investigation. There is no easy way to update the private/public CA key pair and make sure all nodes are informed.

--***MOCA*** [29] [30]: MOCA is basically an extension to Z-H [23]. The focus is on distributed CA services and communication between the nodes and the server nodes – MObile Certificate Authorities (MOCAs). Whereas Z-H does not state how to select CA servers, MOCA suggests the nodes that exhibit best physical security and computational resources should serve as MOCAs. The MOCA scheme furthermore "moves" the combiner function of Z-H from the CA servers to the requesting end-nodes. The benefit is a less vulnerable scheme as the nodes no longer depend on the availability of the CA server nodes to combine the partial certificate signatures.

A MOCA certification protocol, *MP*, is proposed to provide efficient and effective communication between clients and MOCAs. According to MP, certificate requests should be unicast to β specific MOCAs that, based on fresh routing entries or short distance, are likely to be accessible. With the (k,n) threshold scheme, k MOCAs are required to complete a certification service. To increase probability of receiving at least k responses: $\beta = k+\alpha$. When availability drops, the protocol returns to flooding (as in Z-H). It is assumed that MP maintains its own routing tables and co-exists with a "standard" ad hoc routing protocol.

Placing the CA servers in more protected nodes fits with the organization of emergency and rescue operations. Whereas the rescue teams typically move on foot, the on-site rescue management is normally vehicle mounted nearby. The rescue management represents a common point of trust. These nodes may be better protected and less resource constrained than the ordinary nodes. However, the comments regarding the Z-H focus on CA availability and applicability for SAD and MAD operations applies to MOCA as well. The MP maintaining its own routing tables in parallel with a "standard" ad hoc routing protocol is superfluous and a waste of bandwidth.

--***Secure and Efficient Key Management (SEKM)*** [31]: In essence, SEKM suggest the servers of MOCA form a multicast group. The aim is efficient updating of secret shares and certificates. A node broadcasts a certificate request to the CA server group. The server that first receives the request, generates a partial signature and forwards the request to an additional $k+\alpha$ servers (not a true multicast). Only k partial signatures are required. The additional ones are for redundancy in case some are lost or corrupted. SEKM does not state how a server can tell it is the first to receive the refresh request and start the $k+\alpha$ forwarding. On the whole, SEKM has the same features as MOCA. The required number of servers still has to be contacted, and the partial signatures returned.

--***Ubiquitous Security Support (UBIQ)*** [32]***:*** UBIQ is a fully distributed threshold CA scheme. Similar to the partially distributed CA schemes Z-H, MOCA, and SEKM, it relies on a threshold signature system with a (k,n) secret sharing of the private CA key. Differently from the partially distributed CA schemes, *all* nodes get a share of the private CA key. A coalition of k 1-hop neighbors forms the local CA functionality. It does not require any underlying routing protocol –

only a node density of *k* or more 1-hop neighbors. Mobility may help finding the required number of CA nodes. UBIQ prescribes share refreshing.

The nodes earn trust in the entire network when they receive a valid certificate. Any node holding a certificate can obtain a share of the private CA key. A new secret share is calculated by adding partial shares received from a coalition of *k* neighbors. The first nodes receive their certificates from a dealer before joining the network. After *k* nodes have been initialized, the dealer is removed. The authors suggest that as the certification service is delivered within 1-hop neighborhoods, some reliable out-of-bound physical proofs, such as human perception, can be used to authenticate new nodes.

Limiting CA service requests to 1-hop neighborhoods is bandwidth efficient and good for the scalability. From a network point of view, the distributed trust management fits with both SAD and MAD operations scenarios. A local coalition can decide to let in nodes from different domains. A drawback in the rescue operations scenario is the possible requirement of human involvement. In addition, *k*, should be chosen carefully. A low value reduces intrusion tolerance. A large *k* necessitates many neighbors. Ref. [33] suggests more shares per node to succeed also with less than *k* neighbors. In effect, this solution gives little else than reducing the value of *k*. Distributing the CA functionality boosts the availability of private key shares. Anyone capable of collecting *k* shares or more can reconstruct the private CA key. Like any public key scheme relying on a trusted entity, there is no easy way to change the private/public CA key pair during operation.

In [34] it is argued that UBIQ may succumb to a Sybil attack [35] where a single node takes on more identities. With off-line authentication of new nodes and the certificates serving as proof of trustworthiness, this is hardly a realistic threat – at least not in settings like emergency and rescue operations. Secure and efficient revocation is an unsolved challenge.

--***Autonomous Key Management (AKM)*** [27]: AKM provides a self-organizing and fully distributed threshold CA. With few nodes in the network, the scheme is parallel to UBIQ. Each node receives a share of the private CA key. As the number of nodes increases, a hierarchy of key shares is introduced. New nodes then receive a share of a share of the private CA key.

The root CA private/public key pair is bootstrapped by a group of neighbor nodes through *distributed verifiable secret sharing* [36]: Each of the *n* neighbors, chooses a secret value S_i, and distributes secret shares of this to the other neighbors using a *(k,n)* secret sharing scheme[2]. Authentication is added off-line. The sum of the individual secret values $S=(S_1+S_2+S_3+..+S_n)$ represents the private CA key. The corresponding public CA key equals g^S (operations are mod prime p). Assuming the nodes publish the individual public values, g^S_i, the public key can be derived without revealing the private CA key by multiplying individual values g^$S= g$^$S_1 * g$^S_2 *...* g^S_n. Fig. 3 shows the principles.

The nodes (N1-N6) and their shares, *f(Ni),* can be regarded as the leaves of a tree-structure. "R" in Fig. 3 is a virtual node representing the private CA key. The probability of a compromise increases with more nodes holding a share of the private CA key. Therefore, when the number of share-holders reaches a certain level, the nodes split into smaller regional groups that set up a new regional key. Before splitting, the nodes N1-N6 hold shares *f(N1) - f(N6)* of the private CA key. Assuming the nodes N1-N3 decide to form a new group and N4 -N6 another; N1 distributes a share of its secret share *f(N1)* to the other nodes in the new group. The others do the same with their key shares. The new regional secret of N1, N2, and N3 equals the sum of their shares S'= *f(N1)+ f(N2)+ f(N3)*, represented as virtual node G in Fig. 3.

When the number of shareholders in any region reaches the specified level, the region is split. Regions are also merged. With less than *k* nodes, there are too few nodes to provide CA service. Certificates signed with regional keys have less assurance then those signed with the CA key. A high-assurance certificate requires partial signatures from nodes in different regions. The scheme assumes the network evolves from the nodes that initiated the AKM service.

MAD operations require nodes from one domain to be included in the other as key-share trees rooted in different private CA keys cannot be merged.

[2] This approach is contributory in nature. However, derivation of the individual private/public key pairs of the nodes is not. AKM is therefore classified as a distributive scheme.

In AKM, each node maintains a CRL. AKM does not specify network wide dissemination of revocation information. A certificate is revoked when at least *k* neighbors have posted accusations against it. From a security point of view, it is questionable to what extent a certificate signed by the private CA key should be revoked by a group holding only a share of the private CA key.

AKM increases intrusion tolerance at the price of communicational cost. Nodes are assumed to disassociate with the previous region and associate with the new when they move from one region of the network to another. Implicitly, the nodes must maintain a view of the key hierarchy and be able to detect regional boundaries. With mobile nodes and unstable links, it is not evident how this can be implemented. The scheme requires the nodes to collaborate on changes in regions and key hierarchy. Byzantine or faulty nodes may delay these operations. In scenarios like emergency and rescue operations, where the CA services primarily are needed for issuance of initial certificates and revocation, a hierarchical AKM with several regions represents a waste of bandwidth. For robustness and scalability, a single region is preferable. The scheme then equals UBIQ.

--***Self-organized Key Management (PGP-A):*** In [34], Capkun, Buttyán, and Hubaux propose a fully self-organizing key management scheme (PGP-A) – a PGP [37] scheme adapted to ad hoc networks. The CA functionality is completely distributed. All nodes have equal roles. They generate their own private/public key pair and issue certificates to the nodes they trust. Certificates are stored in the nodes rather than in centralized repositories. PGP-A assumes trust is transitive, i.e. if Alice trusts Bob, and Bob trusts Charlie, then Alice should also trust Charlie. The nodes merge their certificate repositories, and try to find a *verifiable chain* of certificates. The *Maximum Degree* algorithm is suggested to construct a certificate graph with high connectivity even if the sizes of the users' certificate repositories are small – due to the *Small World* phenomenon (the hypothesis that everyone in the world can be reached through a short chain of social acquaintances). Certificates are revoked through revocation messages from their issuer, or implicitly revoked at expiry time. Renewals require contact with the issuer. Certificates are also exchanged periodically between neighbor nodes. Evaluation of expiration times and periodical exchanges requires some sort of synchronization between the nodes. It is not evident from the paper how this synchronization should be established.

The periodic certificate exchanges and contact with issuers to have certificates updated is bandwidth consuming and scales badly. PGP-A implicitly requires an already running routing protocol. Trust could be established ad hoc through physical contact and key-exchange via a side channel. This enables improvisations suitable for both SAD and MAD operations. However, human interaction to keep network service running is undesirable.

Byzantine behavior or faulty nodes have limited power to prevent others from exchanging certificates. A compromised node only discloses the keys held by this node. Still, a compromised node could be used to issue certificates allowing other illegitimate nodes to gain access to the network.

There is only a probabilistic guarantee that a chain of trust can be found between parties wishing to communicate. On the other hand, trust transitivity combined with the reliance on the small world phenomenon implies that everyone will soon end up trusting everyone. The result is no intrusion resistance. An alternative would be to restrict the maximum number of hops, and allow the nodes to differentiate the level of trust they put in the various certificates, as suggested in COMP[38].

--***Composite key management for ad hoc networks (COMP):*** COMP [38] combines MOCA's [29][30] partially distributed threshold CA with PGP-A[34] certificate-chaining. The aim is higher security than obtainable with PGP-A, and increased availability of the CA service compared to MOCA. Nodes that have been certified by the CA are allowed to issue certificates to others. Nodes requesting a certification service should first try the MOCA CAs. If this fails, they should search for neighbors that have been certified by the CA. Depending on configuration, nodes with longer certificate chains to the CA may also be entitled to issue certificates to others.

Each certificate in COMP includes a *confidence value* reflecting the level of confidence the certificate issuer has in the binding between node identity and key (0=no trust, 1=full trust).

Multiplication of the confidence values gives a measure for the level of trust in a certificate chain. Short certificate chains are generally preferred over long ones. The probability of one or more compromised nodes in the chain grows as the length of the chain increases. Similarly to PGP-A, COMP assumes a level of trust transitivity. However, signing a certificate, verifying you believe a key belongs to a certain identity, does not necessarily have to mean you also trust this identity to correctly sign certificates of others.

The confidence values enable fine grained evaluation of trust, and the nodes do not have to trust the CA fully. However, deciding a proper confidence level is difficult. COMP does not state how the certificate issuers should accomplish this. Byzantine or compromised nodes may in any case assign full trust to untrustworthy nodes. Nevertheless, intrusion tolerance is increased compared to pure PGP-A as COMP restricts the maximum length of the certificate chains.

Off-line authentication typically includes human interaction, which is cumbersome in the setting of emergency and rescue operations. Interaction with one neighbor is less demanding than the UBIQ requirement for involvement of several neighbors, though. Still, COMP scales no better than MOCA as nodes requesting CA service should first try the MOCA CAs. Transfers of certificate chains limit the scalability additionally.

In MANETs for applications like emergency and rescue operations, the CA will be expected primarily to be needed to issue and revoke certificates. Periodical updates of the certificates should not take place on-line during a rescue operation. Revocation is not addressed by COMP. It is reasonable that the node that issued a certificate is entitled to revoke it. But empowering single ordinary nodes to revoke certificates issued by the CA solely because they hold a certificate signed by the CA, renders the system vulnerable to compromised and Byzantine behaving nodes. Allowing a single node to issue certificates contradicts the purpose of the distributed CA.

A search for neighbors certified by the CA in order to obtain an initial certificate requires knowledge of the public CA key. Hence, at some point there should have been an authenticated channel between the searching node and the CA. The initial authenticated channel is typically obtained through physical contact, or a short range side channel. A natural question for the node asked to provide CA service is then: Why did the requesting node not receive its certificate through the authenticated channel simultaneously?

In SAD operations the certificates could be pre-distributed. In MAD operations it may be hard for a node to verify whether a certificate from the other domain really has been certified by the correct CA or not.

--***Mobility-based key management scheme (MOB):*** MOB [39][40] seeks to mimic human behavior: if people want to communicate securely, they just get close to each other in order to exchange information. Security associations are established between pairs of nodes that get close. The scheme can be fully self-organizing (***MOB-so***) or rely on an off-line authority (***MOB-a***). MOB-so can be based on symmetric or public keys. MOB-a is intrinsically public key based.

A major difference between MOB-so and MOB-a lies in the level of human involvement. In MOB-so, the users should authenticate the communicating peer physically before they establish a security association. The security credentials, *triplets*, are then exchanged over a secure (short range) side channel. The triplets include *user identifier*, *key* and *node address*. The nodes also sign and exchange a statement that proves a security association has been established between the two. MOB-so accepts one level of transitivity in trust: security associations can be established through *friends*, i.e. nodes that have security associations to both nodes in question. MOB-a assumes pre-distributed certificates, and suggests the exchange of security credentials is restricted to 1-hop neighborhood.

In both MOB-so and MOB-a, only the keys held by the specific node are compromised when a node is captured. Byzantine behavior or faulty nodes do not inhibit others from exchanging security credentials. The off-line authority assumed by MOB-a implies no revocation. The authors suggest compromised nodes should revoke their own certificates. However, it can be hard to tell whether a compromise has taken place or not. Revocation on suspicion represents a vulnerability. It may be a threat to availability. Furthermore, if the node has been captured, it may no longer operate according to protocol. With MOB-so, it is left for the user to decide which of its security associations are no longer valid and what friend nodes have turned into enemies.

The MOB schemes are bandwidth efficient in the sense that security credentials are only exchanged within 1-hop neighborhoods. Still, the scalability is limited. The MOB schemes imply a long delay to establish security associations with all communication partners. This is also unsuitable for emergency and rescue operations.

The designers suggest MOB-a for routing security and lower layers, and MOB-so for the application layer. MOB-a fits the SAD operations setting. MAD operations would require an on-line certificate authority to distribute cross-certificates of the merging domains.

MOB-a brings little achievement over pre-distributed certificates without restrictions on certificate exchanges. Depending on routing protocol, confining certificate exchanges to 1-hop neighborhood may inhibit efficient network formation. There is no security achievement from such a restriction. The signature of the authority ensures the validity of the certificate no matter from whom the certificate was received. The assumption of MOB that no one should communicate securely with parties they have not been close to, contradicts the evolution of PKI.

--Identity-based public key (IBC-K): *Identity-based Cryptography* [5], introduced by Shamir, removes the need for certificates. Identities are typically short – at least compared to certificates with a size of several kilobytes. Assuming information that is by default transferred in the routing messages can be used as the public key, identity based schemes may scale better than the traditional certificate based approaches. This makes Identity-based protocols interesting for bandwidth limited ad hoc networks.

Shamir constructed an identity-based *signature* (IBS) scheme. To verify a signature, it is enough to know the ID of the sender plus the *public system parameters*. The public system parameters are defined by the private key generator (PKG) during system set up. The public system parameters include the public key of the PKG and information about the message space. The PKG also generates the private signature keys corresponding to the user IDs. Fig. 4 shows a sketch of Shamir's IBS scheme. During the *setup* phase, the PKG, chooses a secret master key and generates the corresponding *public system parameters*. Afterwards, in the *extraction* phase, it issues *private keys.* The private keys are uniquely given by the IDs and the PKG private master-key.

Several IBS schemes have later been proposed. Some examples are found in [41], [42], and [43]. Boneh and Franklin [44] introduced the first practical identity-based *encryption* scheme (IBE). This scheme has later been extended by Lynn [45] to provide message authentication at no additional cost. The ciphertext itself serves as the message authentication code.

Integration of identity-based signature and encryption schemes (IBSE) is studied in [46]. The latest progresses in IBE encompass strengthened security. Boneh and Boyen [47] suggested the first IBE scheme proven to be secure also in security models without random oracles. Waters suggests a more efficient version in [43]. However, the IBE, IBSE and IBSC schemes presuppose pairwise communication. None are applicable for network layer one-to-many signing and verification of routing information.

The PKG represents a single point of failure. If the private master key of the PKG is compromised, the entire system is compromised. To counter this, Boneh and Franklin [44], suggest spreading the PKG master-key over more locations using threshold cryptography.

Khalili, Katz and Arbaugh [48] propose a key management technique (IBC-K) for ad hoc networks combining identity-based cryptography with threshold cryptography [25]. The nodes that initialize the ad hoc network form a threshold PKG, spreading the PKG private master key over the initial set of nodes by a (k,n) threshold scheme. This eliminates the PKG as a single point of failure, and adds intrusion tolerance. It makes the service robust in the sense that an adversary must compromise minimum k nodes in order to recover the secret master key. It also reduces vulnerability as the service is available as long as k correctly behaving PKG nodes are within reach.

In order to receive the private key corresponding to some identity, a node must present its identity to k (or more) of the n PKG nodes. The node receives a share of the private key from each of them. With k correct shares, the node can then compute its personal private key.

In SAD operations the private keys and system parameters could be handed out from an off-line PKG in the preparation phase. The IBC-K approach fits both SAD and MAD operations scenarios as it makes self-configuration of the

PKG service possible. However, off-line and mutual authentication is in any case required between entering nodes and any PKG node issuing a private key or key share. A secure channel is required. This implies physical contact or a short range dedicated communication channel. Multi-hop connectivity is not good enough. It would enable passive eavesdropping as well as man-in-the-middle attacks.

When time is scarce, physical interaction with a number of geographically distributed PKG nodes is no good solution. Hence, for scenarios like emergency and rescue operations, a single PKG, e.g. located at the on-site rescue management, or a hierarchy of PKGs [49], would be more acceptable.

Explicit key revocation remains an unsolved problem. There is no easy way of distributing revocation lists (withdraw IDs) and make sure all nodes are informed. Another alternative is to change the PKG master key and system parameters. All private keys are derived from these parameters. In essence, an update of the PKG key makes all keys in the system obsolete.

--***Public Key Schemes – Summary:*** The capabilities of the public key schemes are summarized in TABLE II. IBC-K, making certificate exchanges superfluous, is an interesting candidate for ad hoc networks. The reliance of a PKG makes it best suited for SAD operations. Depending on whether the security policy demands centralized trust management or not, IBC-K or COMP/UBIQ fits better in case of MAD operations.

2) *Symmetric Schemes*

--***Pre-shared group key (PSGK):*** This is the old and well-proven key management scheme with a key distribution centre pre-distributing a symmetric key to all members of the group. A key distribution centre could also provide pairwise unique keys, but the focus here is on group keys. The symmetric group key can be used to “sign” routing information with a cryptographic checksum – MAC (Message Authentication Code).

PSGK lacks intrusion tolerance in the sense that security succumbs to a single captured node. But if the security policy allows it, it is a simple solution. Assuming an off-line key distribution center and pre-distributed keys, the scheme scales well. It is immune to faulty nodes and Byzantine behavior. MAD operations would require a means of transferring the group key from one node to another (via a location limited optical channel or similar). Authentication should be added off-line. With a single group key, there is no easy way to exclude compromised nodes.

PSGK was not designed specially for ad hoc networks. It is included here as several of the symmetric schemes studied basically represent extensions to this scheme.

--***SKiMPy*** [50]***:*** SKiMPy is designed for MANETs in emergency and rescue operations. It seeks to establish a MANET wide symmetric key for protection of network layer routing information or application layer user data. On MANET initialization, all nodes generate a random symmetric key and advertise it within 1-hop neighborhoods through HELLO messages. The *best* key, i.e. the one with lowest ID number, freshest timestamp or other, is chosen as the local group key. The best key is transferred to the nodes with worse keys through a secure channel established with the aid of pre-distributed certificates. The procedure is repeated until the “best” key has been shared with all nodes in the MANET. Once established, the group key serves as proof of trustworthiness. SKiMPy proposes periodical updates of the group key to counter cryptoanalysis. The updated keys are derived from the initial group key.

SKiMPy is bandwidth efficient in the sense that nodes agree on the best key locally. There is no need for an already running routing protocol as the key information is exchanged between neighbors only. SKiMPy implies a delay to spread the best key to all nodes. Still, the currently best local key can be used to communicate securely until the “ultimate” key is received.

Byzantine behavior or faulty nodes may disturb local key agreement, e.g. by announcing a better key but not responding.

SKiMPy is designed for SAD operations. MAD operations would require some means of spreading cross-certificates. The authors indicate persons with special roles or ranks could be empowered to administer certificates. However, on-line revocation is not possible before the network has been initialized. As the network is initialized, the symmetric group key is also established. Once the symmetric key has been received, there is no efficient way to expel the node from further participation. The group key (or a key derived from it) now serves as proof of trustworthiness. Thus, SKiMPy adds complexity compared to PSGK, but does not increase the security accordingly.

--***Self-Healing session key distribution (S-HEAL)*** [51]: S-HEAL is a symmetric group key distribution scheme with revocation, designed for networks with unreliable links. The concept demands pre-shared secrets and a group manager that broadcasts the current group key K "masked" with a polynomial $h(x)$; $f(x)=h(x)+K$. Individual secrets $h(i)$ are pre-distributed *(i* refers to *node ID).* Each member node can then extract the current key by evaluating the received expression at $x=i$, and subtracting the secret value; $f(i)-h(i)=K$. All operations take place in a finite field F_q where q is a prime larger than the number of nodes.

Revocation is enabled by replacing the polynomial $h(x)$ with a bivariate polynomial $s(x,y)$. The group manager now broadcast the current key K masked as $f(N,x)=s(N,x)+K$. In order to extract the key, the nodes must first recover the polynomial $s(x,i)$ and evaluate it at $s(N,i)$. Then they must subtract the result from the received $s(N,x)+K$, evaluated at $x=i$; $K= f(N,i)-s(N,i)$.

The thought is that only non-revoked nodes shall be able to recover the polynomial $s(x,i)$. Given s of degree t, $t+1$ values are required to find $s(x,i)$. The value N and the individual secrets, $s(i,i)$ are pre-distributed. The other t values, $s(r_1,x), s(r_2,x)... s(r_t,x)$, that are required to reveal $s(x,i)$, are incorporated in the key update message from the group manager. If the revoked nodes are included in the set $\{r_1, r_2...r_t\}$, these nodes will only acquire t of the required $t+1$ values. Consequently, they will not be able to extract the new group key. The scheme enables revocation of maximum t nodes.

A main feature of S-HEAL is its *self-healing* property. Nodes that lose one or more key distributions can still reveal the missed keys. Each key update message includes shares of all of earlier as well as all possible future keys. The key shares received *before* are complementary to the shares received *after* the key has been distributed. Assuming $p(x)$ is the share received *before* K is distributed, the share received in key update messages *after* K has been distributed equals $K-p(x)$. Hence, missed keys can be derived by combining shares received before the lost update with shares received after the lost update. Whereas the self-healing feature may be of great value in mail systems and similar applications, network layer routing information has only instant value. Hence, retrieving earlier keys is of little interest. Further details are therefore left out.

S-HEAL's reliance on a group manager – possibly multi-hops away – to provide the initial group key, makes it inapplicable for protection of routing information. The group key is needed in order to bootstrap the network service, but S-HEAL demands an already running network service to distribute the group key. Nevertheless, S-HEAL could potentially be used for revocation and re-keying, assuming a protected network service has been bootstrapped with an initial pre-distributed group key (PSGK). It would improve intrusion tolerance compared to pure PSGK. Robustness to packet losses could be increased by periodically retransmitting the latest key update rather than waiting for the next key update as implied by [51].

Regarding scalability, the message sizes and number of key update messages are independent of the number of nodes in the network. The size of the key updates is only proportional to the size of the polynomials (if self-healing is left out.)

Pre-distributed individual shares are acceptable for SAD operations, but incompatible with merging of nodes from different security domains. Thus, S-HEAL has limited applicability in MAD operations.

Missing source authentication of the broadcasts from the group manager is a shortcoming. A MAC generated by the previous group key could easily be added. Still, a Byzantine behavior node could potentially transmit garble, claiming to be the next key from the group manager, and cause disruption.

--***Logical Key Hierarchy (LKH):*** Group keys can be updated brute force: A group manager distributes the new group key, encrypted with a separate (individual) key for each node. In essence, LKH represents a family of schemes that improve the scalability of this brute force method by organizing the keys into a logical hierarchy and giving the nodes additional keys.

LKH was introduced by Wong, Gouda, and Lam [52] and Wallner, Harder, and Agee [53]. The concept is illustrated in Fig. 5. All group members (N1-N8) possess the group key $K_{12345678}$. The sub-group key K_{1234} is shared by members

N1- N4, and K_{12} is common to N1 and N2. K_1 – K_8 refer to the individual keys. Assuming node N8 is to be revoked; all group and sub-group keys known to N8 ($K_{12345678,}$ K_{5678} and K_{78}) should be updated. N7 shares all intermediate keys from the leaf to the root with N8. N7 must therefore receive the updated keys encrypted with its individual key. The new group key and sub-group key can be distributed to N5 and N6 encrypted with their key in common; K_{56}. To N1-N4, the group manager sends the new group key $K_{1234567}$ encrypted with K_{1234}. Thus, bandwidth and computational cost is saved compared to updates encrypted with the individual keys.

The key tree can be binary or k-ary, and balanced or unbalanced.

Whereas the basic LKH scheme was not designed specifically for ad hoc networks, Rhee, Park and Tsudik [54][55] suggest a LKH scheme for hierarchical ad hoc networks. They propose that the group manager functionality is distributed over several managers, each controlling different *cells* of the network. The approach is cellular and infrastructure based rather than ad hoc. The nodes are fully dependent upon the cell managers. Each cell has a different group key.

The nodes must contact the cell manager to receive the key when they move from one cell to another. In other words, the nodes must be able to detect cell boundaries and be within communication range of a cell manager. In addition, the scheme requires that cell managers communicate during "key hand-off" from one cell to another. The intention of the scheme is to limit re-keying to part of the network. The price is reduced robustness and increased bandwidth consumption. The scheme is inapplicable for MANET use.

A number of other refinements of the basic LKH scheme [52][53], focusing on communicational and computational cost, have been proposed. OFT[56][57], OFC[58], ELK[59], LKH+[56], EBHT[60], LKH++[61], Poovendran and Baras [62], and the Internet-draft by Selcuk, McCubbin and Sidhu [63] all propose different ways to reduce communication overhead –primarily focusing on message sizes. Of these schemes, only LKH++ is designed for wireless networks.

ELK reduces the size of the key update messages by sending only part of the key plus a key verification value. The receivers must search brute force for the remaining part of the key. The verification value is used to decide whether the correct key has been found or not. LKH+ and EBHT suggest new keys are derived by applying a one-way function to the old key(s) when members are added. In OFT, OFC, and LKH++ the keys of parent nodes are related to keys of their child-nodes through one-way functions. After a group change, the group manager only sends enough information to enable the nodes to compute the rest of the updated keys themselves. Poovendran and Baras [62] show that the overhead can be reduced by placing nodes that are most likely to be revoked as close as possible to the root, i.e. giving them only a minimum number of keys. Similar ideas are also studied in [63]. In a practical situation it can be hard to decide which nodes are most likely to be compromised. Furthermore, in ad hoc networks, the *number* of messages may be more devastating than the *size* of the messages [64]. Hence, the actual gain of the proposed refinements is not evident. Simulations are required in order to judge which approach is the better.

LKH++ claims to reduce the number of messages as some of the nodes will be able to calculate the new key by themselves. The keys of parent nodes are related to keys of their child-nodes through one-way functions. According to LKH++ and referring to Fig. 5, K_{12} equals a hash of K_1, and K_{1234} represents a hash of K_{12}. Consequently, children to the left will be able to calculate the new key of their parent node. The others receive it from the group manager. Still, the nodes to the left must also be made aware that a key update is required. LKH++ does not address how.

In an ad hoc network, re-keying every time a new node join or leave the network is unnecessary and undesirable. Routing information has only instant value. Backward and forward secrecy on joining and leaving nodes is of little importance. However, LKH may be of interest as an extension to PSGK for revocation purposes. Assuming the network service has been initialized (with the aid of the pre-distributed group key); LKH could be used to expel compromised nodes. A static tree, large enough to hold the keys of all anticipated members, would be required in order to avoid re-keying when new nodes are added. This may be possible for SAD operations. MAD operations would require the merge of two (or more) trees and necessitate re-keying.

For (infrequent) revocations in ad hoc networks for emergency and rescue operations, robustness is even more important than communicational and computational cost. In the basic LKH scheme, innocent nodes that miss the update from the group manager may be cut off. Periodical retransmissions of the last update(s) could help. In ELK, the group manager sends repeated hint messages that enable nodes that lost the key update to calculate the key. Forward error correction codes (FEC) on key updates, as suggested by Wong and Lam [65], enable correction of bit errors, but does not help nodes that missed the entire update.

The group manager represents a single point of failure. Replication of the group manager for reliability and performance, as suggested in [52], is of limited value for ad hoc networks. Replication demands synchronized servers and increase the number of targets for security attacks.

A general weakness of schemes that rely on symmetric keys only, is the missing possibility for source authentication. Byzantine nodes may pose as group manager and cause disruption. Ref. [52] proposes authentication through digital signatures. The basic key management problem then reverts to public key distribution.

--***Probabilistic Key Pre-distribution (PRE):*** PRE [66] assumes WSN nodes outfitted with a pre-installed *key ring*, i.e. a set of keys drawn randomly from a large pool of keys. When bootstrapping the network, the nodes broadcast the identifiers of the keys in their key ring. A wireless link is established between nodes only if they share a key. Hence, resilience to Byzantine behavior and faulty nodes is fine. The scheme relies on a controller node (base station) to broadcast a signed list of the key identifiers to be revoked.

A number of probabilistic key pre-distribution schemes for WSNs have been proposed. In [67], Chan, Perrig, and Song suggest extensions to [66] that increase the resilience against node capture. It requires q common keys ($q>1$) instead of just a single one to establish a connection. Liu and Ning [68] propose probabilistic pre-shared *polynomials* for establishment of pairwise keys in WSNs. Polynomial sharing increases resilience to captured nodes. A trusted entity defines a bivariate polynomial, $f(x,y)$ with the property $f(x,y)=f(y,x)$. Secret polynomial shares; $f(i,y)$ are pre-distributed to each sensor node, i. Any two nodes, i *and* j, can set up a pairwise key by evaluating the polynomial at $f(i,j)$ and $f(j,i)$, respectively. Similarly, Du, Deng, Han and Varshney [69] suggest another scheme relying on probabilistic pre-shared polynomials for pairwise keys in sensor networks. In [70], Du et al. suggest use of deployment knowledge to increase the probability that two nodes find a common secret key. The latter may be possible in a WSN with planned positioning of sensors, but not in a MANET.

Zhu, Xhu, Setia and Jajodia [71] propose probabilistic key pre-distribution combined with secret sharing to set up pairwise exclusive keys in MANETs. A node, wishing to communicate securely with another, picks a secret symmetric key. It then sends shares of this secret symmetric key, encrypted with different pre-distributed keys to the opposite party, i.e. the shares are sent through different logical paths. Assuming the aggregated set of pre-distributed keys used are known to the two nodes in question only, no other nodes will be able to decrypt enough shares to reveal the secret symmetric key. Depending on configuration, the scheme may produce a large number of messages. Reference [71] claims it is desirable to trade computation for communication in ad hoc networks. This assumption does generally not hold.

The idea of the key ring of PRE is intrusion tolerance. The price is availability. There is only a probabilistic assurance that a node actually will share a key with one or more neighbors and be able to bootstrap communication. Emergency and rescue operations, where availability is a number one concern, would require a key ring large enough to achieve close to zero probability of failure. The consequence is intrusion resistance reduced to a level comparable to a pre-distributed group key (PSGK). This contradicts the intension of the scheme. The applicability and scalability for network layer security is limited. Different keys in common with the various neighbors imply more signatures for each routing message. End-to-end signatures on routing messages to be flooded are precluded.

-- ***Security Protocols for Sensor Networks (SPINS)*** [72]***:*** The SPINS security protocols for WSNs assume pre-installed individual (pairwise) keys between the sensor nodes and a base station. Nodes that want to communicate securely, request the base station for a common key. The base station returns the key, encrypted with their individual keys. This scheme demands an already running routing protocol and reliable access to the base station. It is inapplicable for the purpose of protecting routing information in a traditional MANET.

SPINS also includes a scheme for authenticated broadcast; *µTESLA*, and describes how this can be used to provide an authenticated routing protocol for sensor networks. µTESLA relies on a pre-distributed *commitment*, i.e. the last key of a one-way key chain, and delayed disclosure of subsequent keys in the key chain. The key chain can be derived by repeated hashes of an initial random key. The key used at time *i* equals a hash (or similar one-way function) of the key used at time *i+1*. The commitment enables the nodes to verify that later disclosed keys originate from the claimed source; repeated hashes of the disclosed key should return the commitment. To send an authenticated packet, the sender computes a message authentication code (MAC) with a key that is secret at that point in time. The receiver stores the message until the key later is disclosed. The nodes must be loosely time synchronized and know the key disclosure schedule. Otherwise, adversaries could forge packets as the receiver would not know whether the key used to calculate the MAC of an incoming packet had been disclosed or not.

The SPINS authenticated routing protocol discovers routes from the nodes to the base station with the aid of µTESLA key disclosure packets flooded from the base station. The sender, from which a node first received the valid µTESLA packet, is set as parent node in the route to the base station. The pre-distributed commitment enables the nodes to verify that the received packet originated from the base station.

This may work for communication from sensor nodes to a base station. The same technique cannot be used in a traditional MANET with a scattered communication pattern. One possibility would be to pre-load all nodes with commitments of the key chains of all other nodes. This would allow any node to authenticate the messages from any other node. Intrusion tolerance would be fine, and robustness to Byzantine behavior and fault nodes is good. However, the nodes would have to be loosely time synchronized and know the key disclosure schedule of all other nodes. The solution would give little flexibility and scale badly. MAD operations and late registered nodes in SAD operations would be precluded. Furthermore, delayed key disclosure is problematic in the setting of mobile nodes and rapidly changing network topology. Altogether, the SPINS key management scheme and authenticated routing protocol is inapplicable for protection of routing information in traditional MANETs.

--***GKMPAN*** [73]***:*** GKMPAN is designed for secure multicast in ad hoc networks. It is basically a revocation and re-keying scheme for PSGK, founded on PRE [66] and µTESLA[72][3]. GKMPAN assumes a pre-distributed *group key* plus a pre-distributed *commitment*. The group key is used to protect multicast communication. The commitment is used for authentication of revocation messages from the *key server*. In addition, GKMPAN assumes each node is equipped with a pre-installed subset of symmetric keys drawn from a large key pool.

In contrast to PRE, the keys in the key set are determined from the ID of the node. On revocation, the key server issues a revocation message containing the ID of the revoked node. All keys in the key set of the revoked node should be erased or updated. Any node can automatically tell from the ID which keys to revoke. The revocation message also identifies a key that is not in the key set of the revoked node, to be used as “update key.”

A new group key is derived from the old one with the aid of a keyed one-way function; the old group key is used as data input and the “update key” as key input. The output is the new group key. Nodes that have the “update key” in their key set, can calculate the new group key without assistance. The others receive it from their parent node, encrypted with one of the (non-revoked) keys in their key set. It is distributed through a multicast tree rooted at the key server. The validity of the revocation message cannot be checked before the key server later discloses the key that was used to compute the message authentication code.

[3] Actually, [73] claims *TESLA*[74] is used for authentication. At the same time [73] assumes a pre-distributed commitment, and states that only symmetric key techniques are used – which implies *µTESLA*. The difference is subtle. µTESLA relies on a pre-distributed commitment. TESLA assumes the initial packet is authenticated with a digital signature.

In order to avoid potential disruptions, the old and new group key should co-exist until all nodes have received the new group key. However, there is no easy way to make sure all nodes have received the new group key. Byzantine behavior or faulty nodes may inhibit efficient exclusion and re-keying. The reliance on a key server and time synchronization are other vulnerabilities. GKMPAN scales fairly in the sense that new group keys can be calculated by the nodes themselves or transferred locally. In addition, GKMPAN increases intrusion tolerance compared to PSGK as it enables node exclusion. The price is reduced availability. Innocent nodes may be expelled if all their keys happen to be in revoked key sets. This is not acceptable in settings like emergency and rescue operations.

--***Secure Pebblenets (PEBL)*** [75]***:*** Pebblenets refer to large ad hoc networks where the nodes are called *pebbles* due to their small size and large number, e.g. WSNs. The aim of PEBL is protection of application data. It establishes and updates a network-wide traffic encryption key, TEK. At the network layer a pre-installed group key guarantees the authenticity of a pebble as a member of a group. Hence, PEBL can be regarded as an extension to PSGK. The assumption is that only nodes possessing the group key are capable of encrypting and decrypting HELLO messages correctly. Furthermore, PEBL assumes the pebbles organize into clusters of 1-hop neighbors. Each cluster selects a clusterhead node. The clusterheads establish a backbone, and compete to become the *key manager*. The key manager generates the traffic key, TEK, which is intended for encryption of application layer data traffic. The TEK is distributed from the key manager to the regular nodes through the clusterheads. It is updated periodically. Each TEK update is preceded by a re-clustering and new selection of clusterheads. This rotation of the clusterhead role is to avoid exhaustion of the nodes acting as clusterheads, and to account for mobility. Nodes that were one-hop neighbors when the cluster was formed may have moved out of the neighborhood.

Nodes that do not behave according to the protocol may disturb cluster formation and TEK updates. PEBL offers no protection against replay or intrusion. PEBL security succumbs to tampering. Both network layer HELLO messages and TEKs are all protected by keys derived from the group key. Anyone possessing the group key will be able to participate in the TEK updates. PEBL in its entirety, with cluster formation and periodic TEK updates, is bandwidth consuming, demands synchronization and makes availability assumptions that renders it not suited for MANET use.

--***Key Infection (INF)*** [76]***:*** INF is intended for WSNs. The scheme assumes static sensor nodes and mass deployment. INF sets up symmetric keys between the nodes and their 1-hop neighbors. The security is based on surprise: It relies on the assumption that during the network deployment phase, any attacker is only able to monitor a fixed percentage of the communication channels. At bootstrap time, every node simply generates a symmetric key and sends it in the clear to its neighbors. A *key whispering* approach is used, i.e. the key is initially transmitted at a low power level. The transmission power is then increased until the key is heard by at least one of its 1-hop neighbors and a reply is received. INF is simple, self-organizing, and robust to Byzantine behavior and faulty nodes. It is bandwidth efficient, and scales well. However, the security is weak. INF is vulnerable to eavesdropping during key whispering. In addition, there is no authentication of the communicating parties. INF's "security through surprise" fails for MANETs where static nodes and instant mass deployment is no option.

--***Localized Encryption and Authentication Protocol (LEAP)*** [77] is designed for static WSNs. LEAP suggests different keys for different purposes. It requires a number of pre-distributed keys. Pre-distributed *individual keys* are used for communication between sensor nodes and the base station. A pre-shared *group key* is applied for protection of broadcast information from the base station. A pre-installed network wide *initial key, K*, is used to derive *pairwise keys* for secure communication between 1-hop neighbors.

During neighbor discovery immediately after deployment, each node *n* derives its master key, K_n. The master key is derived as a function of its node ID and the initial key; $K_n=f_K(ID_n)$. The master key is used to "sign" HELLO messages. Any node that knows the initial key is able to calculate the master key of any other node ID. Hence, each node can verify the HELLO messages received from its neighbors. The node then calculates the pairwise keys shared with its neighbors, *v*, as a function of their master key and the node ID; $K_{nv}=f_{Kv}(ID_n)$.

Intrusion tolerance is obtained under the assumption of stationary nodes; the network key is erased after the pairwise keys have been established. Nodes that have erased the network key can no longer establish pairwise keys. New nodes can still be added though. As the new nodes have not yet erased the group key, they can set up pairwise keys with their neighbors. When a node is captured, only the keys held by the captured node are compromised.

The pairwise keys are used both to secure ordinary data and to distribute *cluster keys*. The cluster keys are employed for secure local broadcasts. Any node simply generates a cluster key and sends it to all neighbors, encrypted with the respective pairwise keys.

Whereas LEAP may work in a static sensor network, the heart of this key management scheme – the setup of pairwise keys – will not work in a traditional ad hoc network. Deletion of the initial key is incompatible with mobile nodes and constantly changing network topology. Evaluating the scalability of LEAP in MANETs makes little sense, as pairwise key set up is precluded after the initial key has been erased.

--Distributive symmetric Schemes-Summary: TABLE III gives an overview of the capabilities of the distributive symmetric key management schemes. The WSN key management schemes generally assume static nodes, mass deployment, node-to-base station communication pattern or are designed to establish pairwise keys. Their aim and assumptions render them inapplicable for protection of routing information in traditional ad hoc networks with mobile nodes. PSGK, or PSGK extended with S-HEAL or LKH for revocation, appear to be the most promising alternatives of the symmetric schemes.

IV. CONCLUSIONS

We find it useful to classify key management schemes for MANETS (mobile ad hoc networks) as either *contributory* or *distributive*. Fig. 6 provides an overview of the schemes surveyed in this paper. Distributive schemes based on symmetric techniques are either intended for traditional MANETs or for wireless sensor networks (WSN). Distributive key management based on asymmetric cryptographic techniques may take on the standard distinction between certificate-based and identity-based schemes.

We were not able to select one single scheme that is intrinsically superior to the others in our comparative work. A general observation is that none of the proposed key management schemes for MANETs are truly effective for all MANET scenarios. The application must be taken into consideration at current state-of-art. There is a lack of reported attention to the challenges presented by the concrete limitations of communication capacity in MANETs. The performance evaluations tend to be restricted to computational complexity considerations, whereas the practical constraint of MANETs is more likely the communication capacity rather than the computational power and energy consumption. Both the size and the number of messages are important. The number of messages can have more impact than the size due to overhead introduced in lower communication layers.

The optimal combination of bandwidth efficiency and robustness against link loss under a given power consumption should be sought in future key management proposals. Also, secure and efficient key revocation remains an open challenge in MANETs.

ACKNOWLEDGEMENTS

We thank the anonymous reviewers for all their suggestions and constructive criticism. We also thank Leif Nilsen, Univeristy of Oslo – University Graduate Centre/ Thales Group Norway for valuable discussions and comments on our work. A special thanks to Prof. Loren Olson at the University of Tromsoe for proof-reading the manuscript.

REFERENCES

[1] IEEE-SA Standards Board, “IEEE Std. 802.11i,” *IEEE*, 2004.

[2] ZigBee Alliance, “ZigBee Standard, version 1”, *ZigBee Alliance*, 2004.

[3] IEEE-SA Standards Board, “IEEE Std. 802.15.4,” *IEEE*, 2003.

[4] Bluetooth SIG, “Bluetooth Core Specification Version 2.0 + Enhanced Data Rate”, *Bluetooth SIG*, 2004.

[5] A. Shamir, “Identity-based cryptosystems and signature schemes,” In *Proceedings of CRYPTO '84*, 1984, pp. 47-53.

[6] S.A. Camtepe, and B. Yener, “Key Distribution Mechanisms for Wireless Sensor Networks: a Survey,” *Technical report TR-05-07,* Rensselaer Polytechnic Institute, NY, USA, 2005.

[7] S. Rafaeli, and D. Hutchison, “A Survey of Key Management for Secure Group Communication,”*ACM Computing Surveys,* vol. 35, no. 3, Sep. 2003, pp. 309-329.

[8] K. Fokine, “Key Management in Ad Hoc Networks,” *Master Thesis*, LiTH-ISY-EX-3322-2002, Lindköpings tekniska högskola, 2002.

[9] J.v.d. Merwe, D. Dawoud, and S. McDonald, “A Survey on Peer-to-Peer Key Management for Military Type Mobile Ad Hoc Networks,” *The Military Information and Communications Symposium of South Africa - MICSSA*, 2005.

[10] D. Djenouri, L. Khelladi, and N. Badache, “A Survey of Security Issues in Mobile Ad Hoc and Sensor Networks,” *IEEE Communications Surveys & Tutorials*, vol.7, no.4, Fourth Quarter 2005.

[11] Y.W. Law, “Key Management and Link-Layer Security of Wireless Sensor Networks, Energy-efficient Attack and Defence,” *PhD Thesis*, CTIT Ph.D.-thesis Series, Series number: 1381-3617, CTIT Number:05-75, 2005.

[12] W. Diffie, and M. E. Hellman, “New Directions in Cryptography,” *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, Nov. 1976, pp. 644-654.

[13] I. Ingemarsson, D. Tang, and C. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. 28, no. 5, Sept. 1982, pp. 714-720.

[14] M. Burmester, and Y. Desmedt, “A secure and efficient conference key distribution system,” *Proceedings of EUROCRYPT'94*, 1994, pp. 275-286.

[15] K. Becker and U. Wille, “Communication complexity of group key distribution,” *Proceedings of the 5th ACM conference on Computer and Communication Security*, 1998, p.1-6.

[16] N. Asokan and P. Ginzboorg, “Key agreement in ad-hoc networks,” *Computer Communications*, vol. 23, no. 17, Nov. 2000, pp. 1627-1637.

[17] M. Steiner, G. Tsudik, and M. Waidner, “CLIQUES: A new approach to Group Key Agreement,” *Proceedings of ICDCS'98*, 1998.

[18] M. Steiner, G. Tsudik, and M. Waidner, “Key agreement in dynamic peer groups,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 8, Aug. 2000, pp. 769-780.

[19] Certicom Corp. “MQV: Efficient and Authenticated Key Agreement,” *Code & cipher, Certicom's bulletin of security and crypto graphy," Crypto Column,* vol. 1, no.2, 2004.

[20] L. Chen, and C. Kudla, “Identity Based Authenticated Key Agreement Protocols from Pairings”, *HP Technical Report HPL-2003-25*, 2003.

[21] Y. Wang, “Efficient Identity-Based and Authenticated Key Agreement Protocol,” *Cryptology eprint Archive*, Report 2005/108, 2005.

[22] M. Cagalj, S. Capkun, and J.P. Hubaux, “Key Agreement in Peer-to-Peer Wireless Networks,” *Proceedings of the IEEE*, vol. 94, no.2, Feb. 2006, pp. 467-478.

[23] L. Zhou, and Z.J. Haas,"Securing ad hoc networks," *IEEE Network Magazine*, vol. 13, no.6, Nov./Dec. 1999, pp. 24-30.

[24] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, Nov. 1979, pp. 612-613.

[25] Y. G. Desmedt, “Threshold Cryptography,” *European Transactions on Telecommunications*, vol. 5, no. 4, July 1994, pp. 449-457.

[26] A. Herzberg, Jarecki, Stanislaw, Krawczyk, and Yung, Moti, “Proactive secret sharing or: How to cope with perpetual leakage,” *Proceedings of Crypto'95*, 1995, pp. 339-352.

[27] B. Zhu, F. Bao, R.H. Deng, M.S.Kankanhalli, and G. Wang, “Efficient and robust key management for large mobile ad hoc networks,” *Computer Networks*, vol. 48, no. 4, July 2005, pp.657-682.

[28] T. Pedersen, “Non-interactive and information-theoretic secure verifiable secret sharing,” *Proceedings of CRYPTO'91*, 1991, pp. 129-140.

[29] S. Yi, and R. Kravets, “Key Management for Heterogeneous Ad Hoc Wireless Networks,” *University of Illinois at Urbana-Champaign*, 2002.

[30] S.Yi, and R. Kravets, “MOCA: MObile Certificate Authority for Wireless Ad Hoc Networks,” *Report No. UIUCDCS-R-2004-2502,UILU-ENG-2004-1805, University of Illinois at Urbana-Champaign*, 2002.

[31] B. Wu, J. Wu, E.B. Fernandez, and S. Magliveras, “Secure and Efficient Key Management in Mobile Ad Hoc Networks,” *Proceedings of IPDPS’05*, 2005.

[32] J. Kong, P. Zerfos, H. Luo, S. Lu, and L. Zhang, “Providing robust and ubiquitous security support for mobile ad-hoc networks,” *Proceedings of the Ninth International Conference on Network Protocols (ICNP'01)*, 2001, pp. 251-260.

[33] D. Joshi, K. Namuduri, and R. Pendse, “Secure, Redundant, and Fully Distributed Key Management Scheme for Mobile Ad Hoc Networks: An Analysis,” *EURASIP Journal on Wireless Communication and Networking*, vol. 5, no. 4, Sep. 2005, pp.579-589.

[34] S. Capkun, L. Buttyán, and J.P. Hubaux, “Self-Organized Public-Key Management for Mobile Ad Hoc Networks,” *IEEE Transactions on Mobile Computing*, vol. 2, no.1, Jan.-Mar. 2003, pp. 1-13.

[35] J. R. Douceur, “The Sybil Attack,” *The 1st International Workshop on Peer-to-Peer Systems (IPTPS'02)*, 2002, pp. 251-260.

[36] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, “Secure Distributed Key Generation for Discrete-Log Based Cryptosystems,” *Proceedings of EUROCRYPT’99*, 1999.

[37] P. Zimmermann, “PGP User’s Guide,” *MIT Press*, 1994.

[38] S.Yi, and R. Kravets, “Composite Key Management for Ad Hoc Networks,” *Proceedings of Mobiquitous’04*, 2004.

[39] S. Capkun, J.P. Hubaux, and L. Buttyán, “Mobility Helps Security in Ad Hoc Networks,” *Proceedings of MobiHoc’03,* 2003.

[40] S. Capkun, J.P. Hubaux, and L. Buttyán, “Mobility Helps Peer-to-Peer Security,” *IEEE Transactions on Mobile Computing*, vol.5, no. 1, Jan. 2006, pp.43-51.

[41] A. Fiat, and A. Shamir, “How to Prove Yourself: Practical Solutions to Identification and Signature Problems,” *Proceedings of Crypto ’86*, 1987, pp.186-194.

[42] J.C. Cha, and J.H. Cheon, “An Identity-Based Signature from Gap Diffie-Hellman Groups,” *Cryptology eprint Archive*, Report 2002/18, 2002.

[43] B. Waters, “Efficient Identity-Based Encryption Without Random Oracles,” *Proceedings of Eurocrypt* 2005.

[44] D. Boneh, and M. Franklin, “Identity-based encryption from the weil pairing,” *Proceedings of Crypto 2001*, 2001, pp. 213-229.

[45] B. Lynn, “Authenticated identity-based encryption,” *Cryptology ePrint Archive, iacr (International Association for Cryptological Research)*, 2002.

[46] X. Boyen, “Multipurpose Identity-Based Signcryption – A Swiss Army Knife for Identity-Based Cryptography”, *Proceedings of Crypto 2003, LNCS*, vol.2729, 2003, pp. 383-399.

[47] D. Boneh, and X. Boyen, “Secure Identity Based Encryption Without Random Oracles,” *Proceedings of Crypto 2004, LNCS*, 2004.

[48] A. Khalili, J. Katz, and W.A. Arbaugh, “Towards secure key distribution in truly ad-hoc networks,” *Proceedings of the IEEE Workshop on Security and Assurance in Ad hoc Networks*, 2003.

[49] D. Boneh, X. Boyen, and E.-J. Goh, “Hierarchical Identity Based Encryption with Constant Size Ciphertext,” *Proceedings of EUROCRYPT’05*, LNCS, vol. 3494, 2005, pp.440-456.

[50] M. Puzar, J. Andersson, T. Plagemann, Y. Roudier, “SKiMPy: A Simple Key Management Protocol for MANETs in Emergency and Rescue Operations,” *Proceedings of ESAS’05,* 2005.

[51] J. Staddon, S. Miner, M. Franklin, D. Balfanz, M. Malkin, and D. Deam, “Self-Healing Key Distribution with Revocation,” *Proceedings of the IEEE Symposium on Security and Privacy*, 2002.

[52] C.K. Wong, M. Gouda, and S.S. Lam, “Secure Group Communications Using Key Graphs,” *Proceedings of the SIGCOMM ’98*, 1998.

[53] D. Wallner, E. Harder, and R. Agee, “Key Management for Multicast: Issues and Architectures,” *IETF RFC 2627*, 1999.

[54] K.H. Rhee, Y.H. Park, and G. Tsudik, “An Architecture for Key Management in Hierarchical Mobile Ad-hoc Networks,” *Journal of Communications and Networks*, vol. 6, no. 2, June 2004, pp. 156-162.

[55] K.-H. Rhee, Y.-H. Park, and G. Tsudik, “A Group Key Management Architecture for Mobile Ad-hoc Wireless Networks,” *Journal of Information Science and Engineering*, vol. 21, no. 2, March 2005, pp. 415-428.

[56] D. Balenson, D. McGrew, A. Sherman,”Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization, ” *IETF Internet-Draft, August 25, 2000,.draft-irtf-smug-groupkeymgmt-oft-00.txt*.

[57] D.A. McGrew, and A. T. Sherman, “Key Establishment in Large Dynamic Groups Using One-Way Function Trees,” *IEEE Transactions on Software Engineering,* vol. 29, no.5, May 2003, pp. 444-458.

[58] R. Canetti, J. Garay, G. Itkis, D. Micciancio, and M. Naor, “Multicast Security: A Taxonomy and Efficient Constructions,” *Proceedings of INFOCOMM’99*, 1999.

[59] A. Perrig, D. Song, and J.D. Tygar, “ELK, a new Protocol for Efficient Large-Group Key Distribution,” *Proceedings of IEEE Symposium on Security and Privacy,* 2001.

[60] S. Rafaeli, L. Mathy, and D. Hutchison, “EHBT: An efficient protocol for group key management,”*LNCS*, vol. 2233, 2001, pp.159-171.

[61] R.D. Pietro, L.V. Mancini, S. Jajodia, “Efficient and Secure Keys Management for Wireless Mobile Communications,” *Proceedings of POMC’02*, 2002.

[62] R. Poovendran, and J.S. Baras, “An Information Theoretic Analysis of Root-Tree Based Secure Multicast Key Distribution Schemes,”*LNCS*, vol. 1666, 1999, pp.624-638.

[63] A. Selcuk, C. McCubbin, and D. Sidhu, “Probabilistic Optimization of LKH-based Multicast Key Distribution Schemes,” *IETF Internet-Draft, January, 2000, draft-selcuk-probabilistic-lkh-00.txt*.

[64] E. Winjum, A.M. Hegland, P. Spilling, and O. Kure, “A Performance Evaluation of Security Schemes proposed for the OLSR Protocol,” *Proceedings of MILCOM’05*, 2005.

[65] C.K. Wong, and S.S. Lam, “Keystone: A Group Key Management Service,” *Proceedings of ICT*, 2000.

[66] L. Eschenauer, and V.D. Gligor, “A key-management scheme for distributed sensor networks,” *Proceedings of the 9th Conference on Computer Communication Security (CCS2002)*, 2002, pp. 41-47.

[67] H. Chan, A. Perrig, and D. Song, “Random key predistribution schemes for sensor networks,” *Proceedings of the IEEE Symposium on Security and Privacy, IEEE Computer Society,* 2003, pp. 197-213.

[68] D. Liu, and P. Ning, “Establishing Pairwise Keys in Distributed Sensor Networks,” *Proceedings of CCS’03*, 2003.

[69] W. Du, J. Deng, Y.S. Han, and P. Varshney, “A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks,” *Proceedings of CCS’03,* 2003.

[70] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, ”A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge,” *Proceedings of INFOCOM ’04,* 2004.

[71] S. Zhu, S. Xu, S. Setia, and S. Jajodia, “Establishing pair-wise keys for secure communicaton in ad hoc networks: A probabilistic approach,” *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP'03)*, IEEE, 2003, pp. 326-335.

[72] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, and D. Culler, ”SPINS: Security Protocols for Sensor Networks,” *Wireless Networks,”* vol. 8, no.5, Sep. 2002, pp.521-534.

[73] S. Zhu, S. Setia, S. Xu, and S. Jajodia, “GKMPAN: An Efficient Group Rekeying Scheme for Secure Multicast in Ad-Hoc Networks,” *Proceedings of Mobiquitous’04*, 2004.

[74] A. Perrig, R. Canetti, J.D.Tygar, and D. Song, “Efficient authentication and signing of multicast streams over lossy channels,” *Proceedings of the IEEE Symposium on Security and Privacy,* 2000.

[75] S. Basagni, K. Herrin, D. Bruschi and E. Rosti, “Secure pebblenets,” *MobiHoc 2001*, 2001, pp. 156-163.

[76] R. Anderson, H. Chan and A. Perrig, “Key infection: Smart trust for smart dust,” *Proceedings of the 12th IEEE International Conference on Network Protocols, ICNP’04*, 2004, pp.206-215.

[77] S. Zhu, S. Setia and S. Jajodia, “LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks,” *Proceedings of CSS’03*, 2003.

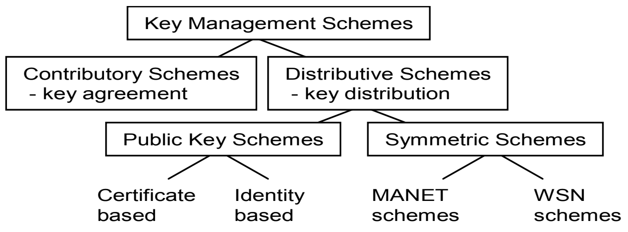

Fig. 1. Classification of key management schemes

TABLE I
SUMMARY OF CONTRIBUTORY KEY MANAGEMENT SCHEMES

		D-H	**ING**	**B-D**	**H&O**	**A-G**	**CLIQ**
Characteristics		Two parties	Logical ring of nodes during D-H key agreement	Reduce the number of rounds to 3 by reliable multicasting	Reduce the number of rounds from *n* to *d* (*n*=2^d) by arranging nodes into hypercube	Password authenticated H&O	Group changes through reliable multicast from group controller
Applicability	Aim	P	G	G	G	G	G
	Net	S. O.	S. O.	Planned	S.O.	S.O.	S.O.
Security	Authentication	N	N	PK	N	Password	N
	Intrusion tolerance	Y	N	N	N	N	N
	Trust Management	Nodes	N	CA	N	Organizer	GC
	Vulnerabilities	MIM	O, MIM, Byz. behavior	O, Byz. behavior	O,MIM, Byz. behavior	O, Byz. behavior	O,MIM, Byz. behavior
Robustness	Availability assumptions	Peer	Ring O	O + RM	Hypercube O	Hypercube O	O+GC+RM
	Byz. behavior & Fault y nodes	Y	N	N	N	N	N
	Group changes	NA	Re-key	Re-key	Re-key	Re-key	Re-key
Scalability		Poor	Poor	Poor	Poor	Poor	Poor

Byz. – Byzantine	*O – node ordering*
G – Group Key (symmetric)	*P – Pairwise symmetric key*
GC – Group Controller	*PK – public key*
MIM – Man-in-the-middle	*RM – reliable multicast*
N – No/None	*S.O. – self organizing*
NA – not applicable	*Y- yes*

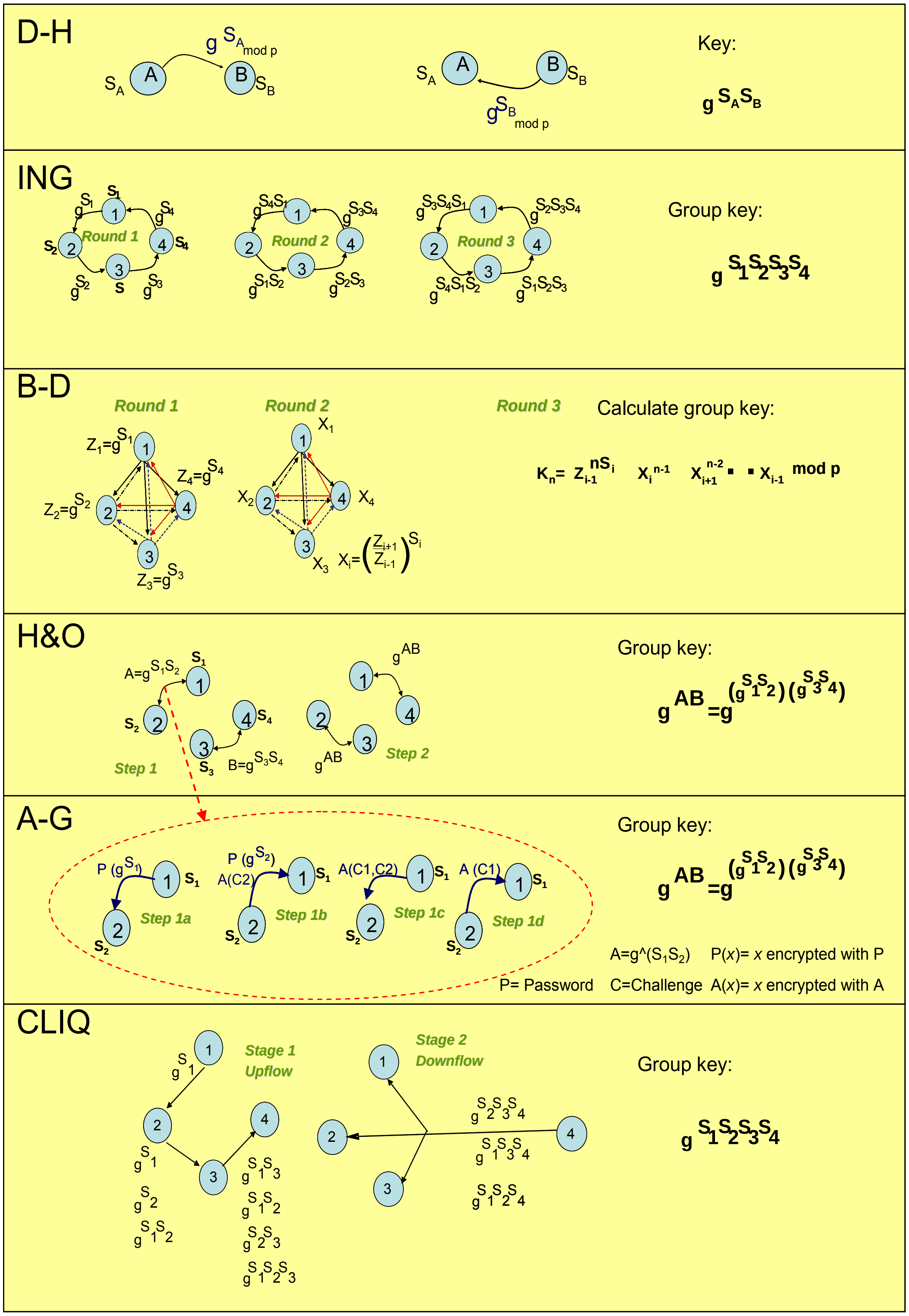

Fig. 2. Outline of the contributory schemes (all exponentiation of generator *g* are modulo prime *p*)

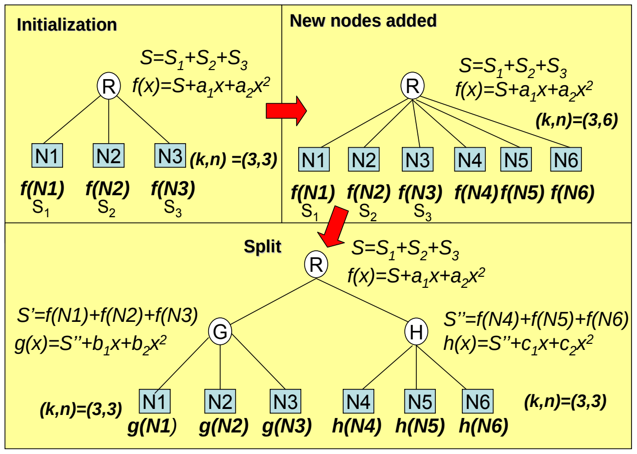

Fig. 3 The principles of secret sharing in AKM; initialization, node addition and regional splits. (All operations are modulo prime *p*.)

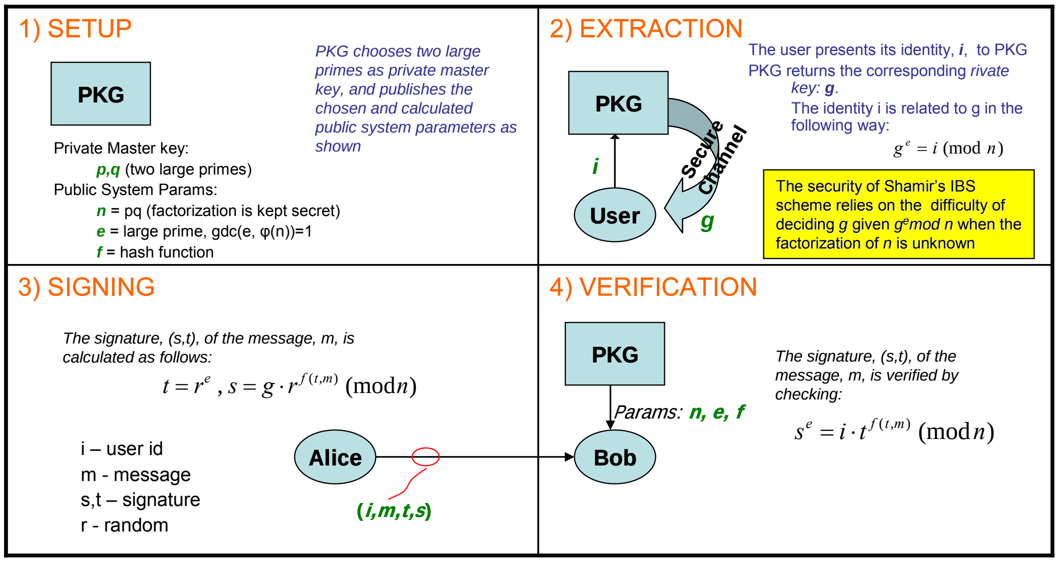
1) SETUP
PKG
PKG chooses two large primes as private master key, and publishes the chosen and calculated public system parameters as shown
Private Master key:
p,q (two large primes)
Public System Params:
n = pq (factorization is kept secret)
e = large prime, gdc(e, φ(n))=1
f = hash function
2) EXTRACTION
PKG
User
i
g
Secure Channel
The user presents its identity, i, to PKG
PKG returns the corresponding rivate key: g.
The identity i is related to g in the following way:
$g^e = i \pmod{n}$
The security of Shamir's IBS scheme relies on the difficulty of deciding g given g^e mod n when the factorization of n is unknown
3) SIGNING
The signature, (s,t), of the message, m, is calculated as follows:
$t = r^e,\ s = g \cdot r^{f(t,m)} \pmod{n}$
i – user id
m - message
s,t – signature
r - random
Alice
(i,m,t,s)
4) VERIFICATION
PKG
Params: n, e, f
Bob
The signature, (s,t), of the message, m, is verified by checking:
$s^e = i \cdot t^{f(t,m)} \pmod{n}$

Fig. 4. Shamir's identity-based signature scheme (IBS)

TABLE II
SUMMARY OF PUBLIC KEY SCHEMES

		Z-H	MOCA	SEKM	UBIQ	AKM	PGP-A	COMP	MOB	IBC-K
Characteristics		Group of servers forms Threshold CA	Most powerful nodes form Threshold CA	Threshold CA servers form multicast group	All nodes part of threshold CA	"Hierarchical UBIQ"	Anarchy: All nodes act as distinct CAs	MOCA+ PGP-A	Move close for exchange of security credentials	No C, Key=ID
Applicability	Aim	PD TCA	PD TCA	PD TCA	FD TCA	FD TCA	FD CA	PD CA	MOB-a: Off-line CA MOB-so: FD TCA	Threshold PKG
	Net	Planned	Planned	Planned	Planned/ Self-org.	Self-org.	Self-org.	Planned/ Self-org	MOB-a: Planned MOB-so: Self-org.	Planned /Self-org
Security	Authentication	Off-line	Off-line	Off-line	Off-line	Off-line	Off-line	Off-line	Off-line/side channel	Off-line
	Intrusion tol.	Good	Good	Good	Fair	Fair/Good	Limited	Limited/Fair	Good	Good
	Trust Mnmt	CA	CA	CA	CA: *k**1-hop neighbors	CA: *k* 1-hop neighbors	Nodes	CA + CA certified nodes	MOB-a: N (Off-line CA) MOB-so: Nodes	PKG
	Vulner-abilities	Combiner, CRL distrib, CA key update	CRL distrib, CA key update	CRL distrib, CA key update	CRL distrib, 1-hop neighbors <k, (Sybil attack), CA key update	Regional changes, Revocation CRL distrib 1-hop neighbors <k, CA key update	Com-promised nodes, CRL distrib,	Com-promised nodes, Distributed trust mnmt, CRL distrib, CA key update	Revocation, Delay due to restriction on Security credential exchanges, CA key update	IRL distrib, PKG key update
Robustness	Avail-ability assumptions	RP, #CA srvrs>k, Combiner, CA srvrs conn., sync	RP, #CA srvrs>k, CA srvrs conn., sync	RP, #CA srvrs>k, CA srvrs conn., sync	#1-hop neighbors> k, sync	#1-hop neighbors from same region> k, Region-awareness	RP, Chains of trust, sync	#CA srvrs>k or CA certified neighbor >1	MOB-a: off-line CA MOB-so: side channel	#PKG nodes > k, PKG node conn., sync
	Byz. behavior & Faulty nodes	Good	Good	Good	Good	Limited	Good	Limited	Good	Good
	Group changes	C + CRL	C + CRL	C + CRL	C + CRL	C+CRL/ accusations + Region size	C + CRL	C (+CRL)	C +CRL	IRL
Scalability		Poor	Limited	Limited	Fair	Limited	Poor	Limited	Limited	Fair

#=The number of
Byz..=Byzantine
C=Certificate
CA=Certificate Authority
CRL=Certificate Revocation List
conn.= Connectivity
distrib=Distribution
FD=Fully distributed
IRL= ID (Key) revocation list
k = threshold value
mnmt=management
N=No/none
PD=Partially distributed
PKG=Private Key Generator
RP=Already running routing protocol
srvrs=Servers
sync=synchronization

TCA= Threshold CA
tol.=tolerance

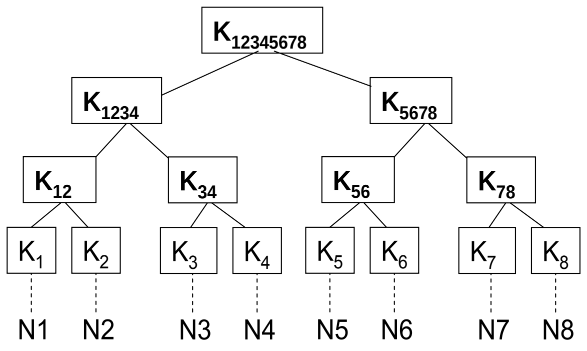

Fig. 5 Logical key hierarchy

TABLE III
SUMMARY OF SYMMETRIC KEY MANAGEMENT SCHEMES

		PSGK	**SKiMPy**	**S-HEAL**	**LKH**	**PRE**	**SPINS**	**GKMPAN**	**PEBL**	**INF**	**LEAP**
	Characteris-tics	Pre-shared group key	Establish key on network formation	Polynomial sharing	Key Hierarchy	Probabilistic key distribution	Suite of protocols for WSN. Key management: Pre-shared keys between nodes and base station	PRE+ μTESLA assisted revocation for PSGK in WSN	Keys for application and network layer - based on PSGK	Whisper key to neighbor	Resists intrusion through non-mobile nodes
Applicability	Aim	GK	GK	Rev & re-key**)	Rev & re-key**)	Keys between subsets of 1-hop neighbors	PK and Authenticated route to base station	GK, rev & re-key	GK	Keys between 1-hop neighbor s	GK, PK, cluster keys
	Net	Plan	Plan	Plan	Plan	Plan	Plan	Plan	Plan	S. O.	Plan
Security	Authenti-cation	Off-line	C, KP	N	KP	KP	Off-line, μTESLA	KP μTESLA	KP	N	KP
	Intrusion tolerance	Poor	Limited	Fair	Fair	Fair	Fair	Limited	Poor	Poor	Limited
	Trust Mnmnt	Off-line	Off-line / special nodes	G Mngr	G Mngr	Controller node	Base station	On-line Key Server	Off-line	N	Base station
	Vulner-abilities	Tamper	Tamper, Rev,CA, Periodic key updates	G Mngr, colluding nodes>t, Byz nodes	GMngr, Byz nodes	Controller node	Base station, Synch	Key Server, synch, Byz nodes, rev of innocent	Tamper, Re-key, synch, Cluster head selection	Eaves-dropping	Initial key, Node mobility, Base station
Robustness	Availability assump-tions	N	N	G Mngr, Reliable key distribution	G Mngr, Reliable key distrib.	Key ring fits with neighbors'	Base station	Key Server	Synch, Full connectivity during TEK establish-ment, Cluster head	1-hop neighbor s> 1	No mobility
	Byz behavior & Faulty nodes	Good	Fair	Poor	Poor	Good	Good	Limited	Limited	Good	Limited
	Dynamic group changes	N	N	Re-key	Re-key	Key re-advertising	N	Re-key	Periodic TEK update	N	Re-key
Scalability	Resource efficiency	Good	Fair	Fair	Fair	Limited	Poor	Fair	Limited	Good	*)

Byz= Byzantine	Mnmt=Management Mngr=manager	rev= revocation
C= Certificate	N=No / none / Not addressed	S.O. = self organizing
CA= Certificate authority	PK= Pairwise keys	synch= synchronization
G=Group	PKI=Public key infrastructure	TEK = Traffic Encryption Key
GK= Group key (symmetric)	Plan = planned	WSN= Wireless Sensor Network
KP= Key Possession	Re-key=re-keying	

*) assumes static nodes – scalability in MANETs with mobile nodes makes little sense
**) It is here assumed a pre-distributed group key and S-HEAL/LKH used for revocation

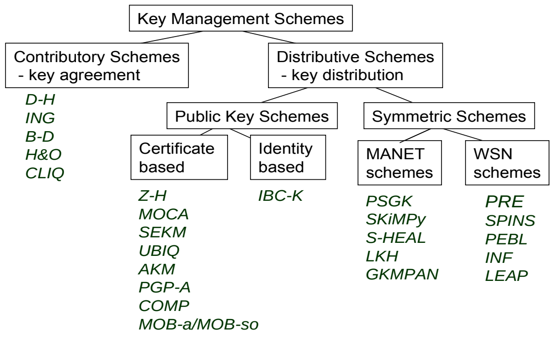

Fig. 6 Key management schemes surveyed