Pre-Loaded Key Based Multicast and Broadcast Authentication in Mobile Ad-Hoc Networks

Mahalingam Ramkumar
Dept. of CSE
Mississippi State University
Mississippi State, MS 39762
ramkumar@cse.msstate.edu

Nasir Memon
Dept of CIS
Polytechnic University
Brooklyn, NY 11201
memon@poly.edu

Rahul Simha
Dept of CS
The George Washington University
Washington, DC 20052
simha@gwu.edu

***Abstract*— The nature of Mobile Ad hoc NETworks (MANET), demands stringent requirements on primitives that could be used to secure such networks. Mobility imposes restrictions on memory and processor requirements due to limited battery life. The ad hoc nature warrants schemes that could operate for extended periods without referring to a Trusted Authority (TA). Additionally, any enabling scheme for security should be able to scale well. We introduce a novel key management scheme, RPS - Random Preloaded Subset key distribution - which satisfies all the above requirements. More specifically, RPS is an n-secure r-conference key predistribution scheme. While most of the previously reported key predistribution schemes [1], [2], [3], [4], [5], also meet all the stringent requirements, RPS has many inherent advantages. In this paper we investigate the applicability of RPS in securing MANETs.**

I. Introduction

MANETs are expected to evolve as the basis for interpersonal communications with perhaps little or no reliance on centralized infrastructure. Such transient networks may be created on demand to facilitate communication between any two nodes (usually) using multiple hops - the nodes en route acting strictly as routers for this purpose. For such applications, nodes engineered to misbehave or act maliciously can wreak havoc on the entire network. To prevent such malicious nodes from taking active part in the network, secure and authenticated communication between nodes is very crucial [6].

In general, the types of communication between various nodes may be classified as unicast, multicast and broadcast. Though multicast communication between r nodes can be achieved by $r-1$ unicast transmissions, multicasting has two obvious advantages - efficient bandwidth usage, and authentication of multicast[1].

Any suitable key distribution scheme for securing MANETS should have the following desirable properties:

1) Ability to operate for extended periods without a TA.
2) No asymmetric crypto primitives due to resource constraints in mobile nodes.
3) Ability to scale well.
4) Instantaneous establishment of session keys.

The first requirement rules out server based key distribution schemes like Kerberos. The second requirement rules out public key cryptography. The third rules out the basic key management scheme using unique pairwise keys. However, all these requirements can be met by key *predistribution* schemes. RPS is a simple key predistribution scheme to facilitate communication between such nodes. RPS permits authenticated secure conference communications between any r nodes without the need for continuous involvement of a TA. In this paper we introduce RPS and study its applicability in r-conference communications and authenticated broadcasts is MANETS (for unicast communications $r=2$).

In Section II we briefly review many n-secure r-conference key predistribution schemes that have been proposed in literature. In Section III we introduce RPS and discuss its applicability in securing conference or unicast communications in MANETS. Section IV addresses various issues in broadcasts performed by nodes, and the use of RPS for broadcast authentication. Conclusions are offered in Section V.

II. n-Secure r-Conference Key Predistribution Schemes

An n-secure r-conference scheme is a systematic method for generation of symmetric keys or secrets, wherein, k (usually different) secrets are preloaded in each node in such a way that any r nodes $u_1,\ldots,u_r \in \mathcal{G}$, can arrive at a session key $K_{\mathcal{G}}$, *independently*. Also, the session key *can not be obtained by any other node*, $a_i \notin \mathcal{G}$, or by coalitions of n nodes $a_1 \cdots a_n \notin \mathcal{G}$. However, there are bound to exist *coalitions* of more than n nodes $a_1,\ldots,a_{n+1},\ldots, \in \mathcal{A}$, $\mathcal{A} \cap \mathcal{G} = \emptyset$ that can jointly arrive at arbitrary group secrets. Pairwise, or unicast communications, can be considered as a special case of r-conferences where $r=2$. As n-secure key predistribution schemes can be (in some case completely) compromised if more than n nodes or compromised, such schemes are generally used with some form of secure hardware.

Any r-conference key predistribution scheme can be generalized as follows. The TA chooses a r symmetric function f and an $r-1$ symmetric functions g_i such that

$$g_i(x_1,\ldots,x_{r-1}) = f(A_i, x_1, \ldots x_{r-1}) \quad (1)$$

The coefficients of g_i are the secrets preloaded in node A_i (node ID A_i). Any node j can independently arrive at r-group

[1] in some cases it may be necessary for *each* of the r nodes to know that *all* the r nodes received the message

keys by substituting the (other $r-1$) node IDs for the variables $x_1 \cdots x_{r-1}$ and evaluating g_j.

In Blom's [1] scheme (proposed for $r = 2$), for n-secure 2-conference (unicast) interactions, a central authority needs to transmit $k = n+1$ elements to each node securely. The TA employs a n degree symmetric polynomial in two variables. The extension of Blom's scheme to multicast security (or for support of conference communications of more than two users in a group) was proposed by Blundo et. al. [3]. For n-secure r-conference, the TA has to generate a symmetric polynomial of degree n in r variables. The TA then has to securely transmit $k = \binom{n+r-1}{r-1}$ secrets to every user.

Matsumota et. al. [2] suggested the generalized model of Eq (1) and used linear symmetric mappings instead of polynomials. In [4], a set of N keys is distributed amongst N nodes. Each node is given a carefully selected set of $n\sqrt{N}$ (for n-secure unicast communication) keys. Two communicating nodes ($r = 2$) use a session key which is based on *all the keys they share*.

The Leighton-Micali [5] scheme is defined by two parameters, k and L, and a cryptographically strong hash function $h()$. The parameter k is the number of keys preloaded in every node, and L is the maximum "hash depth" of the preloaded keys. The TA generates k secret, "root" keys $[M_1 \cdots M_k]$. The one way function $h()$ is used to derive more keys by repeated application of $h()$ on the root keys. The parameter L is the maximum number of times the function $h()$ may be applied. Every node is preloaded with k derived keys, the hash depth of which are determined by the "public key" of the node (which can be tied to it's ID). Any set of r nodes, after exchanging their public keys (or their IDs) can hash each of their k keys forward a certain number of times to reach a common set of k keys from which the session key is derived.

III. Random Preloaded Subsets

In Eschenauer et. al. [7], sensor nodes are preloaded with a randomly chosen set of k keys from a pool of P keys. Two nodes can exchange messages only if they share at least one key. Unlike the other key predistribution schemes, the authors do not assume that the set of preloaded secrets in each node is a function of node ID. In [8], Chan et al. propose a modification of the method by Eschenauer et. al. [7], called the q-composite random key distribution scheme where the sensor nodes need to share at least q keys to form a secure link. If two nodes share q or more keys, *all* shared keys are used to derive the pair key. Both these methods [7], [8] however, cannot be considered as a form of r-conference n-secure schemes - the main reason for this is that the preloaded secrets *are not* tied to the node ID. Also, it is not the purpose of the schemes to facilitate communication between *all* pairs of nodes.

The r-conference n-secure RPS key predistribution scheme draws some ideas from [7], and other key predistribution schemes where the secrets preloaded in a node are tied to its public ID. Like [7], the nodes are preloaded with randomly drawn k keys from a larger pool of P keys. However, unlike [7] and [8], two nodes do not need to go through a series of exchanges to determine the keys they share. A public one way function $F_1()$, determines the "public key" of each node. The public key of node A is

$$[I_1 \cdots I_k] = F_1(A). \tag{2}$$

where, $1 \leq I_1 \cdots I_k \leq P$ is a random permutation of numbers between 1 and P. For instance, it could be obtained by choosing the first k elements of a random permutation of numbers between 1 and P. $I_1 \cdots I_k$ is the index of the keys preloaded in node A. By exchanging IDs, two nodes can immediately determine the shared indices, and use all shared keys to derive pair key. For a r-user conference, the r nodes can independently calculate the conference key based on the keys that *all r nodes share*.

A. Analysis of RPS

- Let $\mathcal{U}_P$ represent a set of cardinality P. Mathematically $|\mathcal{U}_P| = P$.
- Let $\mathcal{A}_k^j$ represent a subset (with index j and cardinality k), of $\mathcal{U}_P$. The set $\mathcal{A}_k^j$ is obtained by *randomly* choosing k elements from the set $\mathcal{U}_P$ without replacement. $\mathcal{A}_k^j \in \mathcal{U}_P$ and $|\mathcal{A}_k^j| = k$
- Let $p_{S_m^{k_1,k_2}}$ represent the probability that the intersection $\mathcal{A}_{k_2} \bigcap \mathcal{A}_{k_1}$ (intersection of two subsets of cardinality k_1 and k_2 respectively) has a cardinality of m.

$$p_{S_m^{k_1,k_2}} = \Pr\left\{|\mathcal{A}_{k_2} \bigcap \mathcal{A}_{k_1}| = m\right\} \tag{3}$$

- $p_{S_m^r}$ represent the probability that the intersection $\mathcal{A}_k^1 \bigcap \cdots \bigcap \mathcal{A}_k^r$ of r sets, has a cardinality m.

$$p_{S_m^r} = \Pr\left\{|\mathcal{A}_k^1 \bigcap \cdots \bigcap \mathcal{A}_k^r| = m\right\} \tag{4}$$

  Let ϕ_r represent the *expected* value of m. Or, $\phi_r = E[m] = \sum_{m=1}^{k} m p_{S_m^r}$.
- Let $p_{C_{n,k}^q}$ represent the probability that the union $\mathcal{A}_k^1 \bigcup \cdots \bigcup \mathcal{A}_k^n$, of n sets has a cardinality q, where $k \leq q \leq q_{max} = \min(nk, P)$.

$$p_{C_{n,k}^q} = \Pr\left\{|\mathcal{A}_k^1 \bigcup \cdots \bigcup \mathcal{A}_k^n| = q\right\}. \tag{5}$$

  Let $\theta_n = E[q] = \sum_{q=k}^{q_{max}} q p_{C_{n,k}^q}$, be the expected value of q;
- Let

$$p_{E_m^q} = \Pr\{\mathcal{A}_m \in \mathcal{A}_q\} \tag{6}$$

  where $\mathcal{A}_m \in \mathcal{U}_P$ and $\mathcal{A}_q \in \mathcal{U}_P$ are arbitrarily chosen sets of cardinality m and q respectively.
- Finally, let

$$p_{E_{\mathcal{R}}}(P, k, n, r) = \Pr\{\mathcal{G} \in \mathcal{B}\}. \tag{7}$$

  where $\mathcal{G}$ is the intersection of r sets $\mathcal{A}_k^1 \bigcap \cdots \bigcap \mathcal{A}_k^r$ and $\mathcal{B}$ is a union of n sets $\mathcal{A}_k^{r+1} \bigcup \cdots \bigcup \mathcal{A}_k^{r+n}$

It can be easily shown that

$$p_{S_m^{k_1,k_2}} = \frac{\binom{k_1}{m}\binom{P-k_1}{k_2-m}}{\binom{P}{k_2}} = p_{S_m^{k_2,k_1}} = \frac{\binom{k_2}{m}\binom{P-k_2}{k_1-m}}{\binom{P}{k_1}} \tag{8}$$

Further,

$$p_{E_m^q} = \frac{\binom{P-m}{q-m}}{\binom{P}{q}} = \frac{(P-m)!q!}{(q-m)!P!}. \quad (9)$$

The probability that r nodes share m keys is represented by $p_{S^r_m}$. Thus $p_{S^2_m} = p_{S^{k,k}_m}$, $p_{S^3_m} = \sum_{i=m}^{k} p_{S^{k,k}_i} \frac{\binom{i}{m}\binom{P-i}{k-m}}{\binom{P}{k}}$, and $p_{S^4_m} = \sum_{i=m}^{k} p_{S^{k,k}_i} \sum_{j=m}^{i} \frac{\binom{i}{j}\binom{P-i}{k-j}}{\binom{P}{k}} \frac{\binom{j}{m}\binom{P-j}{k-m}}{\binom{P}{k}}$. Further, the expression for the probability of a union of n nodes resulting in q unique keys can be expressed, for $n = 1, 2, 3$ as $p_{C^q_{1,k}} = \delta(q-k)$, $p_{C^q_{2,k}} = p_{S^{k,k}_{2k-q}}\ i$, and $p_{C^q_{3,k}} = \sum_{i=0}^{i_{max}} p_{S^{k,k}_{k-i}} * p_{S^{k+i,k}_{2k-q+i}}$, where $i_{max} = \min(q-k, k)$.

Eq (7) can now be written as

$$p_{E_{\mathcal{R}}}(P,k,n,r) = \sum_{q=k}^{q_{max}} p_{C^q_{n,k}} \sum_{m=0}^{k} p_{S^r_m} p_{E^q_m}. \quad (10)$$

Note that the second summation in Eq (10) (over m - or the number of shared keys) starts from 0. This implies that there is a possibility that two (or r nodes) do not share *any* key. In this case, the corresponding eavesdropping probability $p_{E^q_0} = 1$. Thus the assumption is that if nodes do not share a key, an eavesdropper can compromise the communication with probability 1.

For higher n (n ¿ 3) it becomes cumbersome to obtain the exact expression for $p_{C^q_{n,k}}$. To avoid obtaining the exact expression for $p_{C^q_{n,k}}$, we could use a first order approximation of Eq (10), viz.,

$$p_{E_{\mathcal{R}}}(P,k,n) \approx \tilde{p}_{E_{\mathcal{R}}}(P,k,n) = \sum_{m=0}^{k} p_{S^{k,k}_m} p_{E^{\theta_n}_m}. \quad (11)$$

where θ_n, the expected value of q can be obtained by a simple recursive equation $\theta_n = \theta_{n-1} + \frac{k}{P}(P - \theta_{n-1})$, starting with $\theta_0 = 0$.

Similar to the first order approximation for $p_{E_{\mathcal{R}}}(P,k,n,r)$ for large values of n, it is also possible to obtain an approximation for large values of r based on the *expected value* of the cardinality of the intersection of r sets ϕ_r. It can be easily seen that $\phi_r = \frac{k^r}{P^{r-1}}$. Now we can define

$$p_{E_{\mathcal{R}}}(P,k,n,r) \approx \hat{p}_{E_{\mathcal{R}}}(P,k,n,r) = \sum_{q=k}^{q_{max}} p_{C^q_{n,k}} p_{E^q_{\phi_r}}. \quad (12)$$

and

$$p_{E_{\mathcal{R}}}(P,k,n,r) \approx \ddot{p}_{E_{\mathcal{R}}}(P,k,n,r) = p_{E^{\theta_n}_{\phi_r}}. \quad (13)$$

Table I shows the optimal values of the pool size P for various values of k for n = 1,2 and 3. The optimal value is chosen to minimize the probability of eavesdropping $p_{E_{\mathcal{S}}}$, by a collusion of n nodes.

Table II depicts the probability of eavesdropping on multicast communications involving 3 and 4 nodes

TABLE I

OPTIMAL VALUES OF P AND THE CORRESPONDING PROBABILITIES OF EAVESDROPPING FOR VARIOUS VALUES OF k AND n.

k	$n=1$		$n=2$		$n=3$	
k	P	p_{E_R}	P	p_{E_R}	P	p_{E_R}
64	110	3.9e-11	185	1.0e-5	246	0.00049
128	219	1.3e-21	374	1.3e-10	502	2.4e-7
256	437	1.6e-42	748	1.6e-20	1000	5.8e-14
512	874	2.3e-84	1493	2.5e-40	2000	2.1e-27
1024	1748	4.4e-168	2985	4.8e-80	4000	4.4e-54

TABLE II

PROBABILITY OF EAVESDROPPING ON A MULTICAST COMMUNICATION FOR $r = 3, 4$.

	$r=3$			$r=4$		
k	P	p_{E_R}	P/k	P	p_{E_R}	P/k
64	87	1.6e-6	1.359	79	0.00009	1.234
128	173	2.4e-12	1.351	158	8.4e-9	1.234
256	346	5.1e-24	1.351	316	7.4e-17	1.234
512	692	2.4e-47	1.351	632	5.6e-33	1.234

B. Summary of Properties of RPS

The properties of RPS can be summarized as follows:

1) RPS uses only symmetric crypto primitives.
2) The performance of RPS depends on an optimal choice of the ratio $\alpha = \frac{P}{k}$, depending on the value of the number of compromised nodes, n. As n increases, so does the optimal value of α. From Table I it can be seen that the optimal value of α for $n = 1, 2, 3$ are respectively $1.707, 2.92$ and 3.91.
3) As r increases, the optimal value of α reduces.
4) The probability of eavesdropping decreases exponentially with increasing k. Thus if k is doubled, the eavesdropping probability is squared.
5) The value of k needed to maintain an eavesdropping probability is approximately linear in n. If the number of compromised nodes is doubled, the value of k should be doubled to maintain the eavesdropping probability.
6) As long as a non zero number of nodes are compromised, there always exists a possibility that an attacker may be able to eavesdrop on a communication. However, by choosing the parameters carefully, the probability of the event can be made arbitrarily small.
7) There exists a probability that two nodes may not be able to communicate securely with each other as they do not share *any* key. The expressions for the eavesdropping probability takes this situation into account - the term in the summation in Eq (10) corresponding to $m = 0$ reflects this situation. The corresponding eavesdropping probability[2] $p_{E^q_0} = 1$.

[2]The eavesdropping probability is generally not affected much by this term as the probability of not sharing keys is extremely small!

C. Advantages of RPS over Other Key Predistribution Schemes

RPS offers substantial advantages over other existing n-secure r-conference key predistribution schemes.

The efficiency of a key predistribution scheme can be measured in terms of the number of preloaded secrets (k) needed for each node to resist collusion of n nodes. For the schemes in [1], [2] and RPS, k is linearly related to n. However, the methods in [1], [2] use computationally expensive finite field arithmetic to calculate session keys. RPS is able to achieve this without employing finite field arithmetic. It should be noted that for [4], the first key predistribution key proposed without finite field arithmetic, the number of preloaded keys is proportional to $n\sqrt{N}$ where N is the total number of nodes in the system. Another key predistribution scheme which eliminates the need for finite field arithmetic is [5]. However, in [5] k is approximately proportional to n^3.

The methods in [1], [2] are provably secure as long as the number of colluders is less than the "design" value. An increase in the number of colluders results in a complete compromise of the system. The failure of the system occurs catastrophically. On the other hand, for RPS (and [5]) the event of eavesdropping occurs with a certain *probability*. For example, if the parameters of the system are chosen such that the probability of eavesdropping is a value x for a design value of $n = n_0$, for $n < n_0$ the probability of eavesdropping is less than x; for $n > n_0$, the eavesdropping probability is greater than x. Thus degradation in security is graceful, thereby avoiding catastrophic failures.

RPS offers more post-deployment flexibility. In general, schemes with non probabilistic metrics of failure [1], [2] cannot cater for multicast group sizes greater than the design value of r_0. On the other hand RPS and [5] can accommodate greater values of r although at a decreased level of security. Thus RPS can cater for increase in r and n without needing any changes to the deployed secrets, at the cost of reduced security.

RPS also has the capability to be deployed in a hierarchical manner. While [5] provides only vertical hierarchy, RPS can provide a more sophisticated tree-hierarchy.

D. System Renewal

The RPS nodes are capable of operating for *extended* periods without a TA. However, periodic renewal of the system is necessary, and is performed by key updates with TA interaction. For this purpose, each node uses an *additional update* key in conjunction with all the k preloaded keys. Updates can be performed gratuitously, to ensure that compromised keys do not serve the attackers for a long time. Even if all the k keys of a node are compromised, in order to participate in subsequent key updates, the attacker needs the update keys. The update keys are therefore given a very high degree of protection from "sniffing". As long as the update keys can be protected, the attacker can use compromised keys to eavesdrop on communications only till the next round of updates. As keys get renewed, compromised keys are rendered harmless. The update key of every node is also stored in the server (TA) and synchronously updated by both the node and the server after very update. It is possible for the update key to be a password used by the person who owns the node, in which case, the update key need not be explicitly stored in the node.

RPS nodes need to expose only two interfaces - Encrypt() and Decrypt(). With appropriate flags these interfaces can used for both inter-nodal and node-TA interactions. For inter-nodal exchanges, as implicit authentication is provided, a key based authentication may not be needed, and could be optional. For ensuring integrity of the communication a shorter hash (not necessarily cryptographically strong) $H_0(X)$ may be used. Note that even the output of the Decrypt() interface is optional. This may happen especially during key updates. The input to the Decrypt() interface may be a message containing key updates from the TA, resulting in internal changes in the node's key-ring.

$$C_{AB} = Encrypt(ID_B, M, FLAGS) \quad (14)$$
$$X = [FLAGS|ID_A|ID_B|E_{K_{AB}}(M)]$$
$$C_{AB} = [X|\langle H_{K_{AB}}(X)\rangle|\langle H_0(X)\rangle] \quad (15)$$
$$M = Decrypt(C_{AB}) \quad (16)$$
$$C_{\Omega A_n} = Encrypt(ID_A, M, FLAGS) \quad (17)$$
$$Y = [FLAGS|ID_A|E_{K_{\Omega A_n}}(M)]$$
$$C_{\Omega A_n} = [Y|H_{K_{\Omega A_n}}(Y)] \quad (18)$$
$$\langle M\rangle = Decrypt(C_{\Omega A_n}) \quad (19)$$

In the equations above, K_{AB} is the session key for nodes with IDs ID_A and ID_B. M is the plain-text, and C_{AB} the cipher-text. $K_{\Omega A_n}$ is the session key between the node and the TA, which is derived from all the k keys of node A and the update key.

IV. Broadcast Authentication

Because every node shares a key with every other node, broadcast authentication is rendered trivial with RPS. A message M to be broadcast can be appended with a key based hash of the message for every intended recipient. For example, for a broadcast from node A to M other nodes $R_1 \cdots R_M$, the transmitted message C can be obtained as

$$C = [ID_A|M|ID_{R_1}\cdots ID_{R_M}|M|H_{K_{AR_1}}(M)|\cdots|H_{K_{AR_1}}(M)]$$

However, broadcast messages may consume significant bandwidth if the number of nodes in the neighborhood of a node is high. To save bandwidth, the broadcast may be authenticated to just one node in the neighborhood.

$$C = [ID_A|ID_{R_1}|M|H_{K_{AR_1}}(M)]$$

If the authentication fails the node R_1 which received the authentication could broadcast a message indicating failure. The other nodes that received the broadcast would then disregard the broadcast by node A. In general, the node A might have to authenticate the broadcast to L nodes. The L nodes however may not be chosen at will by the broadcasting node. If every node is aware of the nodes in a two-hop neighborhood, then each neighbor of the broadcasting node may be able to

verify a "global rule" for choice of neighbors for authentication, depending on the topology of its neighbors. Figure 1 depicts four different topologies in the neighborhood of node A. For case (i), all neighbors of node A, viz B, C, D, E and F are connected to each other through a path that does not go through A. In this case, A needs to authenticate broadcast only to one of the five nodes B to F. So, A would authenticate to the node with the smallest ID. For case (ii), once again, is similar to case (i) with respect to existence of paths. However, in this case node A would need to authenticate its broadcast to node C (as this minimizes total number of hops to reach all other nodes in case authentication fails)[3]. For case (iii) however all neighbors are not connected by a path[4] that does not rely on A. So A would need to authenticate itself to two of its five neighbors. In this particular instance, A would authenticate itself to nodes B and E or D (whichever has a smaller ID). For case (iv), A would need to authenticate itself to three nodes, F, B/C, and E/D. To reduce the freedom of nodes to collude, it is necessary that the broadcasting node does not have the *option* to choose the recipient(s) of authentication. The choice is dictated by the topology and with the knowledge of all two hop neighbors, every neighbor of the broadcasting node can also independently arrive at the choice of nodes that will receive authentication.

It should be noted that in case (iii) C and D are connected through another node G one hop away from both C and D. Therefore, node A is also aware of the existence of node G. In fact, A is also aware of the fact that G is one hop away from both C and D (and thus serve as an alternate path between C and D). Even though nodes B and E are aware of the presence of node G, they do not know that G is one hop away from *both* C and D. Node F is completely unaware of the presence of G. For the purpose of choosing nodes for authentication A should disregard the existence of node G. In other words, the choice of node(s) for authentication should be based on the knowledge common to *all* receiving nodes.

An inherent disadvantage of using broadcast is the susceptibility to jamming attacks (note that only the broadcast authentication is key based, the broadcast data itself is not). This could be alleviated significantly if broadcast is replaced by a multicast to all the neighboring nodes. In this case, only the exchanges of IDs need to occur before a key agreement for the multicast is reached. Nodes might then tune to a system-wide channel (which may use a single fixed key or an open channel) only for a very small percentage of time to "welcome" new neighbors.

V. Conclusions

We have analyzed RPS, a novel key pre-distribution scheme, for its applicability in MANETs. The computational complexity of RPS would depend on the symmetric crypto primitives for one-way functions used to obtain the session keys from the shared keys. No finite field arithmetic is necessary.

[3]Once again, if there is a tie between multiple nodes, the node with the smallest ID would be chosen.

[4]Ignore node G for the moment.

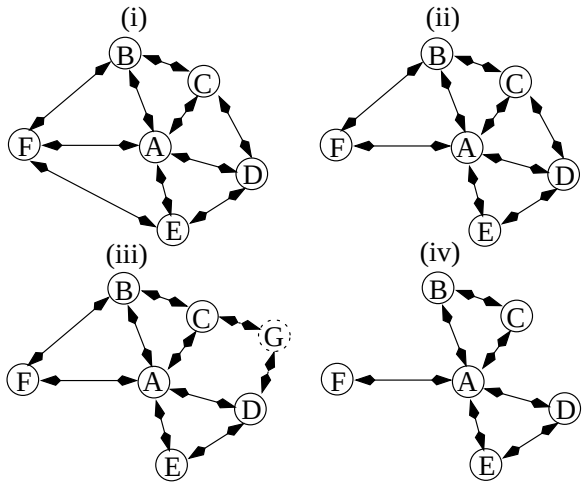

Fig. 1. Four topologies to illustrate broadcast authentication

Because of the probabilistic merit of the security of the scheme, the degradation in security (as more nodes get compromised) is graceful, similar to [5]. Unlike key predistribution schemes with deterministic security merits, RPS and the method in [5] also allow for greater post-deployment flexibility. The maximum size of conferences r can also be increased with some security trade-off. In a MANET setting multicast would be primarily used to replace broadcast to neighbors. In this case one could afford to live with reduced security of multicast. The main advantage of RPS over [5] is that RPS is able to achieve a *linear* relationship between the number of compromised nodes n and the number of preloaded secrets needed k. A recent extension of RPS, which is a generalization of RPS and the Leighton-Micali [5] scheme can be found in [9].

References

[1] R. Blom, "An Optimal Class of Symmetric Key Generation Systems," *Advances in Cryptology: Proc. of Eurocrypt 84*, Lecture Notes in Computer Science, **209**, Springer-Verlag, Berlin, pp. 335-338, 1984.

[2] T. Matsumoto, H. Imai, "On the key predistribution system: A practical solution to the key distribution problem," Advances in Cryptology – CRYPTO '87, vol 293, Lecture Notes in Computer Science, pp 185-193, August 1987.

[3] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, M. Yung, "Perfectly-Secure Key Distribution for Dynamic Conferences," Lecture Notes in Computer Science, vol 740, pp 471–486, 1993.

[4] L. Gong, D.J. Wheeler, "A Matrix Key Distribution Scheme," *Journal of Cryptology*, **2**(2), pp 51-59, 1990.

[5] T. Leighton, S. Micali, "Secret-key Agreement without Public-Key Cryptography,"*Advances in Cryptology* - CRYPTO 1993, pp 456-479, 1994.

[6] M. G. Zapata, N. Asokan, "Securing Ad-Hoc Routing Protocols," Proceedings of the 2002 ACM Workshop on Wireless Security (WiSe 2002), pages 1-10. September 2002.

[7] L. Eschenauer, V.D. Gligor, "A Key-Management Scheme for Distributed Sensor Networks," Proceedings of the Ninth ACM Conference on Computer and Communications Security, Washington DC, pp 41-47, Nov 2002.

[8] H. Chan, A. Perrig, D. Song, "Random Key Predistribution Schemes for Sensor Networks," IEEE Symposium on Security and Privacy, Berkeley, California, May 2003.

[9] M. Ramkumar, N. Memon, "HARPS: HAshed Random Preloaded Subset Key Distribution," Cryptology ePrint Archive, Report 2003/170, 2003, http://eprint.iacr.org/2003/170.