An Efficient Key Predistribution Scheme for Ad Hoc Network Security

Mahalingam Ramkumar and Nasir Memon

***Abstract*—We introduce hashed random preloaded subsets (HARPS), a highly scalable key predistribution (KPD) scheme employing only symmetric cryptographic primitives. HARPS is ideally suited for resource constrained nodes that need to operate for extended periods without active involvement of a trusted authority (TA), as is usually the case for nodes forming ad hoc networks (AHNs). HARPS, a *probabilistic* KPD scheme, is a generalization of two other probabilistic KPDs. The first, random preloaded subsets (RPSs), is based on random intersection of keys preloaded in nodes. The second, proposed by Leighton and Micali (LM) is a scheme employing repeated applications of a cryptographic hash function. We investigate many desired properties of HARPS like scalability, computational and storage efficiency, flexibility in deployment modes, renewability, ease of extension to multicast scenarios, ability to cater for broadcast authentication, broadcast encryption, etc., to support its candidacy as an enabler for ad hoc network security. We analyze and compare the performance of the three schemes and show that HARPS has significant advantages over other KPDs, and in particular, over RPS and LM.**

***Index Terms*—Key distribution, key predistribution, mobile ad hoc network (MANET) security.**

I. Introduction

THE TREND toward decentralization is perhaps the most significant technological change in the last two decades. The Internet is a very good example of how pervasive such a decentralized system can become. This trend is more readily seen in many evolving applications, perhaps fueled by the instant popularity gained by file sharing systems. Many wireless devices are also expected to follow this trend.

Such decentralized ad hoc networks are expected to play a crucial role in every aspect of human life. The ad hoc networks (AHNs), which do not *rely* on any significant infrastructure, are themselves expected to *become* major infrastructures in our day to day lives. It is, therefore, imperative in such applications, to have efficient means of developing *trust* between the nodes of the network. For example, nodes forming mobile ad hoc networks (MANETS) have to perform authenticated exchanges for building a routing table, or relaying messages between other nodes. In such a scenario, malicious action by a single node could have a potentially disruptive effect over the entire network.

Manuscript received October 15, 2003; revised September 30, 2004.

M. Ramkumar is with the Department of Computer Science and Engineering, Mississippi State University, Mississippi State, MS 39762 USA (e-mail: ramkumar @cse.msstate.edu).

N. Memon is with the Department of Computer and Information Science, Polytechnic University, Brooklyn, NY 11201 USA (e-mail: memon@poly.edu).

Digital Object Identifier 10.1109/JSAC.2004.842555

The ability of a node A to *trust* node B could be achieved by a process, that permits node A to verify that node B is "compliant" to a set of pre-imposed rules. This in turn could be achieved by permitting such compliant nodes to establish *authenticated shared secrets* that other (noncompliant) nodes cannot. The secrets that are stored in such compliant nodes (or secrets that only compliant nodes have access to), thus, serve as a "hook" for compliance. The process of establishing such authenticated shared secrets (either from secrets preloaded in each node or secrets that nodes have access to), could be achieved by a suitable key distribution scheme (KDS).

However, designing a KDS for this purpose is very challenging due to the many inherent restrictions in such deployments. The nodes involved are typically battery operated wireless devices with severe resource constraints. Furthermore, the network may consist of millions or even billions of nodes. Additionally, the nodes may not have persistent access to a centralized trust authority (TA).

Severe resource constraints rule out KDSs employing asymmetric cryptography. The need for scalability rules out the "basic" key distribution scheme[1] as a possible choice. The lack of persistent access to a centralized trust authority rules out KDSs like Kerberos, where an active presence of a trusted server is necessary. This leaves us with key predistribution (KPD) schemes.

A KPD consists of a TA and N nodes with unique IDs. A certain number (say k) of secrets, chosen by the TA, are preloaded in each node, such that any two nodes A and B can discover a shared secret K_{AB} *independently*. Also, no *other* node can arrive at K_{AB}. In general, a KPD may also facilitate discovery of shared *group* keys for group sizes of up to g nodes. A KPD that can be used for group sizes up to g and resistant to collusion of n nodes (no coalition of up to n nodes can discover the group secret) is represented as a (g, n)-KPD or a g-conference, n-secure KPD. The k preloaded secrets in each node are collectively referred to as the node's *key-ring*.

KPDs are inherently *tradeoffs between security and complexity*. In fact, the "basic" key distribution scheme is also a KPD. It is a very "secure" KPD—no matter how many nodes are compromised by an attacker, the attacker cannot discover the shared secrets between nodes that have *not* been compromised. This security, however, is achieved at the expense of very high storage complexity—each node needs to store $O(N)$ keys (size of key-ring) for pairwise communications. However, by relaxing security requirements (say by mandating that it

[1]In the basic key distribution scheme ($\binom{N}{2}$ keys are distributed amongst N nodes, and each node gets $N-1$ keys. As the number of keys distributed to each node depends on the total number of nodes in the system, such a scheme does not scale well.

is sufficient for the system to be secure until n nodes have been compromised by the attacker), it is possible to reduce the key-ring size in each node. For example, with many KPD schemes it is possible to achieve this requirement with just $O(n)$ keys in each node (typically, $n \ll N$). The relaxed security requirements can achieve tremendous gains in storage complexity. More importantly, the fact that the storage complexity is not directly related to the network size N (and only dependent on the maximum possible attacker coalition size n), implies that such schemes could be very useful for very large N. In practice, limiting n (the size of attacker coalitions) can perhaps be achieved by some reliance on "read-proof hardware" [1] of the nodes with preloaded keys, and periodic renewal of the preloaded keys.

It is important to realize that for any application scenario involving *mutual cooperation* of nodes (like MANET applications), compromise of a *few* nodes may affect the *entire* system—*irrespective* of the nature of the KDS used to protect the deployment. For if the "owner" of a node has *complete* control over the node, then a coalition of few such owners could (working in tandem) disrupt the entire network. This implies that even the owner of a node should *not* have access to the secrets buried in the node (which guarantee compliance). This in turn implies that read-proof hardware is *mandatory* for such deployments [2]. The dire need for this no doubt will drive technology[2] to improve it—after all, it has always been necessity that drives innovation. Nevertheless, as history has shown, given unlimited time and resources, any form of tamper resistance could perhaps be broken. Therefore, any security solution for deployments of MANET nodes should cater for periodic *renewal* of secrets.

In this paper, we propose a novel KPD scheme, hashed random preloaded subset (HARPS), and investigate its suitability for securing AHNs. While *all* KPD schemes are tradeoffs between security and complexity their basic differences lie in the *actual tradeoff employed*. Almost all KPDs satisfy the three fundamental requirements[3] for securing resource constrained ad hoc networks. HARPS, however, possesses the following very useful additional properties that makes it very useful for purposes of securing AHNs.

1) Obtaining session keys from preloaded secrets is not computationally expensive.
2) The failure (or compromise of security) of the system does not occur *catastrophically*.
3) Efficient in terms of the number of preloaded keys (k) in each node.
4) Easy extension to multicast communication scenarios.
5) HARPS is renewable.
6) Permits hierarchical deployments.
7) Provides mechanism (broadcast encryption) for periodic *revocation* of nodes suspected of being compromised.
8) Provides mechanisms for broadcast authentication.

[2]This is already evidenced by a slew of companies developing tamper resistant/read-proof modules. See for instance eracom-tech.com, spyrus.com.au, ncipher.com, aepsystems.com, rainbow.com, thales-esecurity.com, cryptomathic.com, h20138.www2.hp.com.

[3]Employ only symmetric cryptography, high scalability, and ability to function without an active TA.

The rest of this paper is organized as follows. Section II is a brief discussion of various KPDs which are classified into two categories—deterministic and probabilistic. As a model for both probabilistic and deterministic KPDs, we extend the notion of (g, n) KPD to (g, n, p) KPD, where p is the *probability* that a (g, n) KPD is *insecure*.[4] The probability p is also referred to as the *eavesdropping* probability. In Section III, we introduce HARPS, and an analysis of its "merit" (lower the eavesdropping probability, higher the merit). HARPS is a generalization of two other random KPD schemes—RPS [3] and LM [4]. The analyses readily demonstrate the superiority of HARPS over RPS and LM schemes. In Section IV, we investigate other useful properties of HARPS like resistance to node synthesis, hierarchical deployments, extension to multicast scenarios, renewability, and ability to provide nonrepudiation and revocation.

Recently, another improvement of RPS (which we shall refer to as MBS) has been proposed independently by Du *et al.* [5] and Liu *et al.* [6]. Section V provides a comparison of RPS, HARPS, and MBS schemes. Conclusions are presented in Section VI.

II. Overview of Key Predistribution Schemes

A (g, n, p) KPD is a systematic method for generation of symmetric keys or secrets such that:

1) any group of g nodes $\{G_1 \dots G_g\} = \mathcal{G}$ can discover the shared secret $K_{\mathcal{G}}$ with probability p_d;
2) a coalition of n nodes $A_1 \dots A_n \notin \mathcal{G}$ can discover the same secret $K_{\mathcal{G}}$ with probability p.

The probability p is the eavesdropping probability, and the probability $p_o = 1 - p_d$ is the "outage" probability, which occurs if the group of g nodes *cannot* discover a shared secret. For a KPD scheme to be "secure" even when an attacker has exposed secrets from n nodes, p_o and p should be "very close" to 0. Furthermore, under the condition of "outage," an attacker can compromise exchanges between nodes *even if he has not compromised any other node* (or $n = 0$). In other words, $p \geq p_o$ *includes* the outage probability. More specifically, $p = p_o$ for $n = 0$, and $p \geq p_o$ for $n > 0$.

A. General Model of KPDs

A KPD consists of a TA and N nodes, with unique IDs (say $\mathrm{ID}_1 \dots \mathrm{ID}_N$). The TA chooses P secrets $\mathcal{R}$ (or $|\mathcal{R}| = P$) and two functions $F(\cdot)$ and $G(\cdot)$ such that

$$\mathcal{S}_i = G_i(\cdot) = F(\mathcal{R}, \mathrm{ID}_i) \tag{1}$$

is the set of secrets $\mathcal{S}_i$, $|\mathcal{S}_i| = k$, that are preloaded in node i with ID ID_i. The secrets $\mathcal{S}_i$ can also be interpreted as the *parameters* of a function $G_i(\cdot)$. Any two nodes i and j, with preloaded secrets $\mathcal{S}_i$ and $\mathcal{S}_j$ can discover a unique shared secret K_{ij} as

$$\begin{aligned} K_{ij} &= G_i(\mathrm{ID}_j) = G_j(\mathrm{ID}_i) \\ &= F(\mathcal{R}, \mathrm{ID}_i, \mathrm{ID}_j) = F(\mathcal{R}, \mathrm{ID}_j, \mathrm{ID}_i) \end{aligned} \tag{2}$$

without further involvement of the TA.

There are, however, restrictions on functions $F(\cdot)$ and $G(\cdot)$ in order to satisfy the requirements in (1) and (2). As the shared secret is a function of the IDs, the ability of nodes to arrive at the

[4]The (g, n) KPD is secure with probability $(1 - p)$.

shared secret provides mutual assurances to i and j that the *other* node possesses the necessary secrets $\mathcal{S}_j$ and $\mathcal{S}_i$, respectively, and can, thus, be "trusted." In other words, the shared secret simultaneously provides mutual *authentication* of IDs.

Obviously, the established trust is based on the assumption that no one else, apart from node j has access to the secrets $\mathcal{S}_j$. Typically, if an attacker manages to "expose" secrets buried in a finite number of nodes—say he manages to expose secrets $\mathbb{S}_A = \{\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_n\}$—he may be able use this "knowledge" $\mathbb{S}_A$ to compromise the system (or "fool" other nodes into performing tasks that may be detrimental to the network at large).

In [7], Matsumota *et al.* presented a generalized model of (g, n) KPD's. The TA employs an g-symmetric function $F(\cdot)$, the coefficients of which are the system secrets, chosen by the TA. Each node employs a $g-1$ symmetric function, say $G_A(\cdot)$ for node A (with ID ID_A), such that

$$G_A(x_1, \ldots, x_{g-1}) = F(\mathrm{ID}_A, x_1, \ldots, x_{g-1}). \tag{3}$$

The "symmetry" of the functions $G(\cdot)$ and $F(\cdot)$ manifest themselves as *invariance to permutation* of the variables $x_1, \ldots, x_{g-1}$. Corresponding to each node, the TA evaluates the function $F(\cdot)$ by substituting the ID of that node, resulting in an expression in $g-1$ variables. The expression in $g-1$ variables is the function $G_i(\cdot)$ for node i with ID ID_i.

(g, n, p) KPD can be classified as deterministic or probabilistic. A deterministic or (g, n) KPD is a special case of (g, n, p) KPD, where p can assume only binary values—0 or 1. For probabilistic schemes, $0 \leq p \leq 1$ can take continuous values.

B. Deterministic Schemes

For a n-secure Blom's scheme [8], for $g = 2$ (pairwise secrets), the function $F(x, y)$ is a symmetric n-degree polynomial in two variables in a prime field q. The secrets provided to a node i are the $(n+1)$ coefficients of the polynomial $G_i(x) = F(x, i)$ in one variable. The extension of Blom's scheme to multicast scenarios $(g > 2)$ was proposed by Blundo *et al.* [9], where $F(\cdot)$ is a symmetric n-degree polynomial in g variables. In this case, node i has $\binom{n+g-1}{g-1}$ secrets corresponding to the coefficients of $G_i(\ldots) = F(i, \ldots)$. Matsumota *et al.* [7] presented a linear symmetric scheme as a specific example of their generalized model (3), employing g-symmetric matrices.

C. Probabilistic Schemes

In the Leighton–Micali (LM) scheme [4] the TA chooses k "root" secrets $[M_1 \ldots M_k]$, and a cryptographic hash function $h(\cdot)$. A one-way function (or a random number generator) $F_L(\cdot)$ is employed to generate a stream of k uniformly distributed random numbers, $F_L(\mathrm{ID}_A) = a_1 \ldots a_k, 1 \leq a_i \leq L$, seeded by node ID. The preloaded keys in node A are $[M_1^{a_1} \ldots M_k^{a_k}]$, where M_i^j is obtained by *repeatedly* hashing M_i, j times. The shared key K_{AB} between nodes A and B is then $K_{AB} = h(K_1 \ldots K_k)$, where $K_i = M_i^{\alpha_i}$, and $\alpha_i = \max(a_i, b_i)$ (and $F_L(\mathrm{ID}_B) = b_1 \ldots b_k$). For the LM scheme, the g-symmetric function is the evaluation of the *maximum* of the "hash-depths" for g nodes for each root key (which is obviously symmetric as it is independent of the *order* in which the nodes are chosen).

Many KPD schemes based on preloading a node with a subset of the TA's keys have been proposed in literature. The earlier schemes were based on *deterministic* allocation of keys to each node. The earliest of such methods [10] was not efficient in terms of number of stored keys needed in each node (each node needs $O(n\sqrt{N})$ keys in order to be n-secure). Many other methods were proposed subsequently, [11]–[13], for which $k = O(n \log N)$—most of them motivated by Erdos *et al.*'s seminal work on intersections of finite sets [14]. However, the complexity of the deterministic key allocation algorithm makes it impractical for nodes to discover shared secrets based on their IDs. To overcome this limitation, nodes may need to employ a bandwidth intensive shared key discovery process, where the nodes exchange information regarding the indices of the keys they possess. However, the inability of nodes to ascribe a sequence of keys to a node (based on ID) implies that authentication of node IDs does *not* occur implicitly.

More recently, many authors have proposed similar methods, where the choice of keys for each node is random [15], [16], or pseudorandom [3], [17], [18]. We shall collectively refer to all these methods as random preloaded subsets (RPS). While all RPS-based methods are essentially similar, methods based on random allocation, similar to the methods that employ complex deterministic constructions, need a bandwidth intensive shared key discovery process (which in turn also means that authentication of node IDs is not implicit). The methods which employ *pseudorandom* allocation [3], [17], [18] of subsets on the other hand, provide a simple and elegant way for nodes to determine shared secrets.

In the RPS scheme (based on pseudorandom allocation of subsets), the TA chooses an indexed set of P secrets $[K_1 \ldots K_P]$. A one-way function (again seeded by the node ID) is used to obtain a *partial*[5] random permutation $[A_1 \ldots A_k] = F_R(\mathrm{ID}_A)$. Now, $[A_1 \ldots A_k]$ are the indices 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 their shared key.

For the probabilistic schemes, there is always a possibility that any exchange can be compromised, but the probability of the event can be made arbitrarily small by choosing the parameters appropriately. The failure of probabilistic systems, therefore, is never catastrophic.

At first sight, permitting a finite probability of compromise may seem to be a major disadvantage. In practice, it is not. For example, consider a n-secure *deterministic* KPD, in which the KPD secrets are used to arrive at a shared secret, which has a *finite* number of bits. If, for example, the final shared secret is 64 bits, an attacker has a probability $(1)/(2^{64}) > 10^{-20}$ that he can "pull the secret out of a hat" (or guess the secret in a single attempt). Thus, as long as the probability of failure (or eavesdropping) is less than the bit-security offered by the key length of the final shared key, probabilistic merits are *in no way inferior*. Probabilistic schemes realize this fact, and utilize them advantageously!

[5]Say the first k numbers of a random permutation of $(1 \ldots P), k < P$.

While LM has a finite probability of eavesdropping when one or more nodes are compromised, it does not have an "outage" probability. RPS on the other hand permits a small outage probability (which is typically much smaller than the eavesdropping probability). Once again, as long as the outage probability is small, it is not an issue, but by taking advantage of this fact, RPS is able to achieve significantly better efficiency than LM (RPS achieves $k = O(n)$, while $k > O(n^2)$ for LM, as we shall see in the next section).

III. HARPS

HARPS is defined by three parameters (P, k, L) (these parameters **do not** represent the parameters (g, n, p) used to describe a general KPD), and two public functions as follows:

1) $h(\cdot)$, a cryptographic hash function;
2) $F(\cdot)$, a public-key-generation function.

Each node is assigned a unique ID. The TA chooses P secrets (or "root" keys) $[M_1 \dots M_P]$. From each root key, one can get L "derived keys" by repeated application of the one-way function $h(\cdot)$ (similar to the LM scheme). The jth derived key from the ith root key (obtained by hashing M_i repeatedly, j times) is represented as $K_i^j = h^j(M_i)$, where $1 \le i \le P, 1 \le j \le L$. For a node A with ID ID_A, application of public function $F(\cdot)$ yields a sequence of k-length ordered pairs

$$[(A_1, a_1), (A_2, a_2), \dots, (A_k, a_k)] = F(\mathrm{ID}_A). \quad (4)$$

The first coordinate of the ordered pairs, viz. $(A_1 \dots A_k)$, is a *partial random permutation* of integers between 1 and $P, 1 \le A_j \le P \, \forall 1 \le j \le k$ and $A_j \ne A_i, i \ne j$ (similar to RPS). The second coordinate $(a_1 \dots a_k)$ is a sequence of uniformly distributed numbers between 1 and L (similar to LM). Each node is preloaded with k secrets. The preloaded secrets in any node depend on the node's ID. The key preloaded in node A for instance, is given by

$$\left[K_{A_1}^{a_1} \dots K_{A_k}^{a_k}\right] = \left[h^{a_1}(M_{A_1}) \dots h^{a_k}(M_{A_k})\right]. \quad (5)$$

For a node B with ID, $\mathrm{ID}_B, [(B_1, b_1) \dots (B_k, b_k)] = F(\mathrm{ID}_B)$. With the knowledge of the node ID's, the two nodes A and B can independently arrive at the *indices* of the shared *root* keys as $[s_1 \dots s_m]$ by application of the globally known public-key generation function $F(\cdot)$ (on their IDs). Corresponding to these shared root keys, denote the m derived keys in node A as $[K_{s_1}^{a_1} \dots K_{s_m}^{a_m}]$, and in node $B, [K_{s_1}^{b_1} \dots K_{s_m}^{b_m}]$. Let $d_1 = \max(a_1, b_1), \dots, d_m = \max(a_m, b_m)$. The session key (or the group secret) K_{AB} is then obtained as

$$K_{AB} = h\left(h^{d_1}(M_{s_1}) \,\middle|\, h^{d_2}(M_{s_2}) \,|\, \dots \,|\, h^{d_m}(M_{s_m})\right). \quad (6)$$

Note that the group secret can calculated independently by both nodes. Without loss of generality, let us assume that $d_1 = \max(a_1, b_1) = b_1$. This implies that node A would arrive at the term $h^{d_1}(M_{s_1}) = K_{s_1}^{d_1}$ in (6) by hashing its key $K_{s_1}^{a_1}, (b_1 - a_1)$ times. As one of B's preloaded key is $K_{s_1}^{b_1} = h^{d_1}(M_{s_1}), B$ does not have to hash forward for this particular key. For each shared key, however, at most one of the nodes has to hash forward to reach a common "depth."

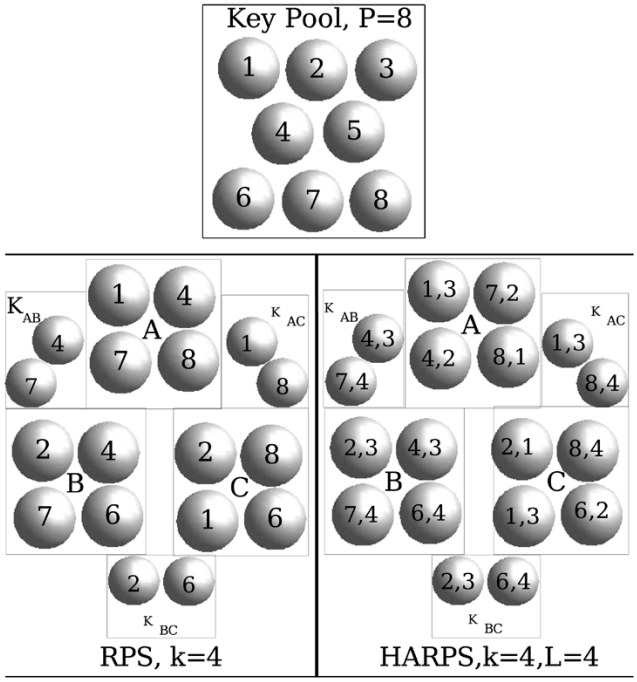


Fig. 1. Illustration of RPS and HARPS for $P = 8$, $k = 4$, and $L = 4$ (for RPS $L = 0$). The TA has a key pool consisting of eight keys $M_1 \dots M_8$. For RPS, the shared secret between nodes A and B is derived from their shared keys $M_4 4$ and M_7. For HARPS, the shared secrets are derived from $K_4^3 = h^3(M_4)$ and $K_7^4 = h^4(M_7)$.

Fig. 1 is an illustration of HARPS for $P = 8, k = 4$, and $L = 4$. In Fig. 1, A and C share root key indexes 1 and 8. For the key indexed 8, A has to hash forward three times to reach the same depth as C. For the key indexed 1, both A and C have the same depth and so no hashing is required of either of them.

HARPS is a generalization of both LM [4] and RPS [3]. RPS is a special case of HARPS for $L = 0$—the keys are *not* hashed before they are preloaded in the nodes. LM is a special case of HARPS for $P = k$—each node has a hashed version of *all* root keys. Fig. 1 also illustrates the similarity and differences between RPS and HARPS.

Network Size: For most KPD schemes, there is practically *no limit on the network size*. For HARPS, the number of possible unique key-rings is $\binom{P}{k}L^k$. As the keys (and, hence, key-rings) are chosen randomly, in order to avoid "collision" (taking "birthday paradox" into account), we could restrict the size of the network $N_{\max}$ to $\sqrt{\binom{P}{k}L^k}$. For example, for $P = 2560, k = 256$ and $L = 64, N_{\max} \approx 1.3 \times 10^{411}$. In practice, the limitation on the size of the network would be based on the *number of bits allocated* for representing the (unique) ID of each node. For a system with 32 bit IDs for instance, the network size would be limited to about 4 billion.

IV. Analysis of HARPS

An analysis of HARPS involves calculation of the eavesdropping probability p [more specifically $p(n, g)$], that an attacker

who *controls a coalition*[6] of n nodes, can discover the shared secret of a group of g nodes.

To analyze[7] the security of HARPS, we shall consider P realizations of a random "experiment." Each instance of the experiment is likened to r users (belonging to a conference group) and n nodes (belonging to an attacker coalition) choosing a *particular* key from the pool of P root keys. For each experiment (corresponding to one of P root keys), the group of g nodes arrive at an *elementary share* of their shared secret with some probability. However, there is also a finite probability that the elementary share is not "safe"—as the n nodes in the coalition of the attacker may (independently or together), be able to arrive at the elementary share.

We shall denote by ε, the probability that the elementary share *is* safe (cannot be discovered by a coalition of n attacker nodes). The final shared secret between the g nodes is derived from *all* P elementary shares. Therefore, in order to compromise the final shared secret, the attacker's coalition should be able to discover *every* elementary share.

Consider an arbitrary key (say index i) from the pool of P keys. Let ξ_{ij} represent the probability that the ith key $1 \leq i \leq P$, is chosen as a candidate for node $j, 1 \leq j \leq N$. As each key is chosen with the same probability, and as any node is equally likely choose any key as a candidate, we have

$$\xi_{ij} = \xi = \frac{k}{P} \forall i, j. \tag{7}$$

The probability that g nodes (belonging to the group trying to discover a shared secret) pick the ith key is ξ^g (g independent draws from a binomial distribution). In other words, ξ^g is the probability that the g nodes arrive at an elementary share. However, the elementary share may not be safe. Out of the n nodes in the attacker's coalition, zero or more nodes may also pick the ith key. The probability that *exactly* $u (0 \leq u \leq n)$ nodes out of n pick the ith key is $B_\xi(n, u)$, where

$$B_\xi(n, u) = \binom{n}{u} \xi^u (1 - \xi)^{n-u}. \tag{8}$$

Obviously, if $u = 0$, the elementary share i is safe. However, even if $u > 0$, there exists a nonzero probability that the elementary share (corresponding to the $i$the key) is safe. Let $\alpha_1 \ldots \alpha_g$ be the hash depths corresponding to the ith root key in each of the g nodes. Let $\beta_1 \ldots \beta_u$ be the corresponding hash depths of the u keys (derived from the ith root key), in the attacker's nodes. The elementary share between the g nodes would be the ith key at hash depth $\alpha = \max(\alpha_1 \ldots \alpha_g)$. As long as $\beta = \min(\beta_1 \ldots \beta_u) > \alpha$, the attacker *cannot* discover the elementary shared secret. We shall the represent the probability $\Pr\{\beta > \alpha\}$ as $\mathcal{Q}(L, g, u)$, or

$$\mathcal{Q}(L, g, u) = \sum_{i=1}^{L} \frac{i^g - (i-1)^g}{L^g} \left(\frac{L - i}{L}\right)^u \tag{9}$$

[6]Mere "ownership" of a node alone is not a sufficient condition for a node to considered as a part of an "attacker's coalition." In order to be part of the coalition, the attacker must have been able to tamper with and *expose* all secrets buried in a node.

[7]The analysis presented in this section is motivated in part by Dyer *et al.* [12], for calculating the probability that an intersection of r sets is contained in an union of n sets.

where $1 \leq u \leq n$. Further, $\mathcal{Q}(L, g, u = 0) = 1$ and $\mathcal{Q}(L = 0, g, u) = 0$. Thus, the probability that the elementary share is safe is

$$\varepsilon = \sum_{u=0}^{n} \xi^g B_\xi(n, u) \mathcal{Q}(L, g, u). \tag{10}$$

Each experiment (or "round") is a face-off between the g-group and the attacker. In order to compromise the shared secret, the attacker has to win all P rounds. The term ε is the probability that an elementary share is safe (or the probability that the attacker loses a round). Therefore, the probability that the attacker wins *any* round is $1 - \varepsilon$. The probability that *all* elementary shares are compromised (or the attacker wins *all* P rounds) is then

$$p(n, g) = (1 - \varepsilon)^P = \left\{(1 - \varepsilon)^{\frac{1}{\xi}}\right\}^k. \tag{11}$$

For the special cases of $\mathrm{RPS}(L = 0)$ and $\mathrm{LM}(\xi = 1)$, the expressions for ε are, respectively

$$\varepsilon = \begin{cases} \varepsilon_R = \xi^g B_\xi(n, 0) = \xi^g (1 - \xi)^n & L = 0, \quad \xi \neq 1 \\ \varepsilon_L = \mathcal{Q}(L, g, n) & \xi = 1, \quad L \neq 0 \end{cases}. \tag{12}$$

Note that the expression for ε for HARPS has $n + 1$ terms corresponding to $u = 0 \ldots n$. For LM ($\xi = 1$), only the term corresponding to $u = n$ is nonzero. For RPS, only the term corresponding to $u = 0$ is nonzero. For the same value of ξ, obviously, $\varepsilon_R \leq \varepsilon$, indicating that the probability that each elemental share is safe is higher for HARPS. Alternately, for the same P, k, the eavesdropping probability for HARPS ($L \geq 0$) is less, or at worst equal, to that of RPS (with $L = 0$), or as L increases, the eavesdropping probability decreases.

Now, let us consider the expression for eavesdropping probability for RPS, for $g = 2$ (pairwise communications). Also, by choosing $\xi = (C)/(n)$, and replacing P with $(k)/(\xi) = (kn)/(C)$, we have

$$p_R(n, g = 2) = (1 - \varepsilon_R)^P \leq \exp\left(-\frac{kC(n - C)^n}{n^{n+1}}\right) \tag{13}$$

which readily indicates that for RPS, for any desired probability of eavesdropping, as n increases, we have to increase k in a linear proportion. In other words, $k \not> O(n)$. Obviously, for HARPS too, as for the same $\xi, \varepsilon \geq \varepsilon_R, k \not> O(n)$.

A. Optimization of Parameters

Optimization of random KPDs is about minimizing the resource utilization in the nodes for a desired level of security. As all three random KPDs have easily achievable computational requirements, the optimization focuses on minimizing k, the size of the key-ring, for a desired level of security (or desired $p(n, g)$)—we are not too concerned about the size of the key pool P. For a given value of k, there are two parameters under the control of the designer—P (or equivalently, ξ), and the maximum hash depth L. Obviously, as value of the hash depth L is increased, p reduces. However, for $L > 64$ the payoff is negligible (as we shall soon see). We shall, therefore, fix the value of L at 64.

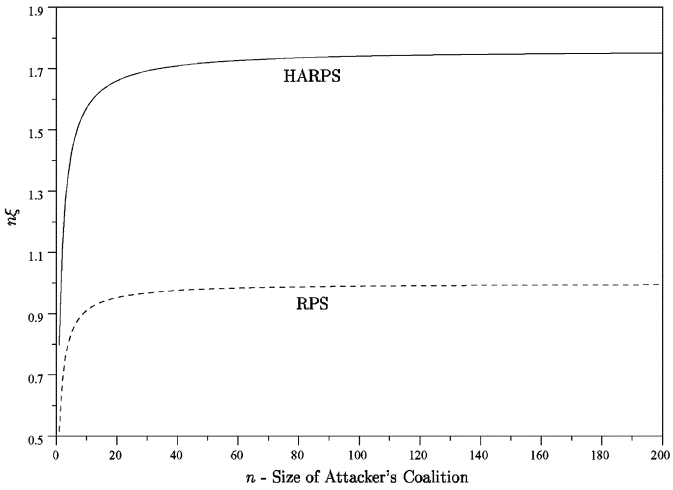


Fig. 2. Plot of ξn versus n for HARPS and RPS. The value of ξ is chosen to minimize the eavesdropping probability for a given k (for HARPS $L = 64$).

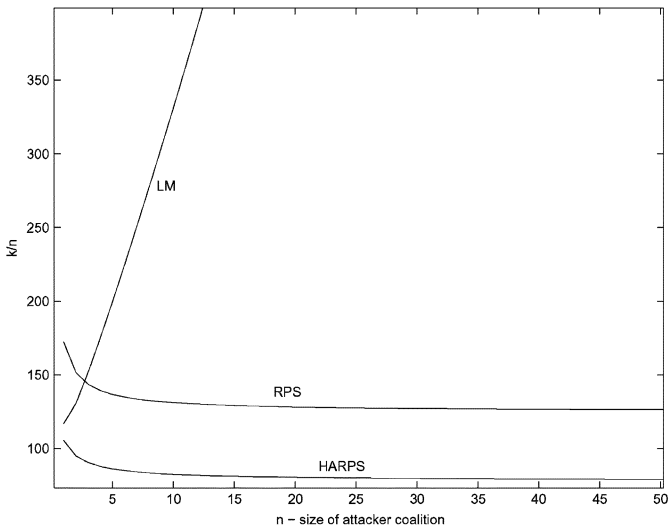


Fig. 3. Plot of k/n versus n for HARPS, RPS, and LM for $p = 10^{-20}$.

From (11), it is readily seen that for minimizing k we need to minimize $(1-\varepsilon)^{(1/\xi)}$. For the LM scheme, there is no scope for optimization as ξ is fixed at one $(k = P)$. For RPS and HARPS on the other hand, an optimal choice of ξ is very crucial. The plots in Fig. 2 indicate the optimal values of ξ for different values of n for HARPS and RPS. It can be easily seen that for both (RPS and HARPS), $\xi \propto (1)/(n)$ ($\xi n \approx 1.7$ for HARPS and $\xi n \approx 0.95$ for RPS).

Fig. 3 is a comparison of HARPS, RPS, and LM in terms of the number of preloaded keys (or the size of the key-ring k) needed to achieve an eavesdropping probability of less than 10^{-20} for various values of the size of attacker's coalition n. The plot of k/n versus n clearly depicts that $k > O(n^2)$ for LM and $k \approx O(n)$ for RPS and HARPS. Specifically, for **HARPS** $k \approx 75n$ and for **RPS** $k \approx 128n$, for $p < 10^{-20}$ and large n. Also, for the same ξ, an increase of k would result in a corresponding *exponential reduction* of p. In other words, $k = -C_1 n \log(p)$.

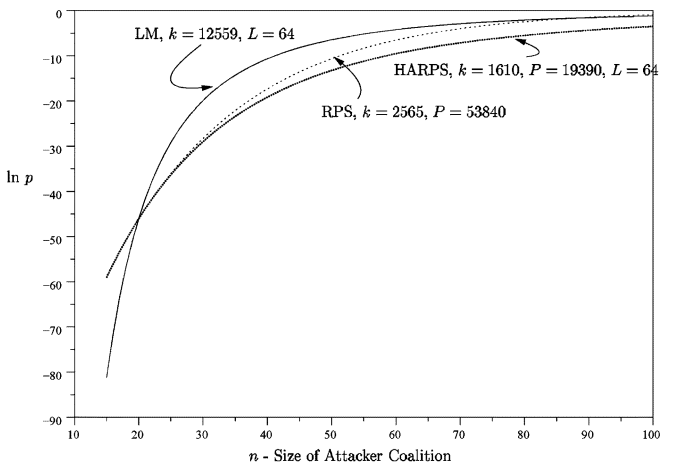


Fig. 4. Performance of HARPS, RPS, and LM designed for $p \approx 10^{-20}$ at $n = 20$ for various values of n. Note the slower degradation of security for HARPS compared to RPS and LM.

It is interesting to note that the constant $C_1 = C_{\mathrm{RPS}} \approx \mathrm{e}$ for RPS, and $C_1 = C_{\mathrm{HARPS}} \approx \sqrt{\mathrm{e}}$ for HARPS.[8]

The relations $k = O(n)$ and $\xi = (k)/(P) \propto (1)/(n)$, imply that $P \approx O(n^2)$ for RPS (and HARPS).[9] For $p \leq 10^{-20}$, this implies $P \approx 42n^2$ for HARPS, and $P \approx 135n^2$ for RPS. However, the dependency $P = O(n^2)$ for RPS and HARPS is not a serious limitation—as P is the *number of keys that the TA needs to store—not the nodes*.

For the plots in Fig. 3, the value of k is calculated by optimizing ξ for each value of n (as shown in Fig. 2). However, optimization is performed *before* deployment, and compromises occur *after* deployment. Thus, for a deployment of any KPD, it is important to know how a system optimized for *some* value of n behaves for *larger values* of n. Fig. 4 is a plot of the logarithm of the eavesdropping probability $(\log(p))$ versus n for HARPS, RPS, and LM systems, all three designed to provide a eavesdropping probability of $p < 10^{-20}$ for $n = 20$ (the intersection of the three curves occurs at $p \approx 10^{-20}$ at $n = 20$). In addition to the *obvious advantage of needing the least value* of k to achieve this requirement, HARPS also has a *slower rate of "degradation of security"* with increasing n. For the systems compared, to compromise internodal exchanges with a probability greater that 0.5, the attacker needs to compromise 220 nodes for HARPS ($P = 19\,390, k = 1610, L = 64$), and only 106 nodes for RPS ($P = 53840, k = 2565$). Also, note that the comparison is not "fair" for HARPS (the comparison is between LM with $k = 12\,559$, RPS with $k = 2565$ and HARPS with $k = 1610$).

Note that $k = 1610$ is the *minimum* value of k needed to ensure an eavesdropping probability $p < 10^{-20}$ for $n = 20$ for HARPS. With a 10% increase in k and maximizing P (resulting in $P \approx 35\,000$) to ensure that $p < 10^{-20}$ for $n \leq 20$, the attacker would need to compromise 300 nodes before he can achieve $p = 0.5$. For a 25% increase in k, the figure goes up to 371 nodes (a 70% increase from 220).

[8]Based on numerical evaluations. We have not been able to *analytically* establish this relationship yet.

[9]This can also be readily seen from (13) by replacing k with $\xi P = (\mathrm{CP})/(n)$.

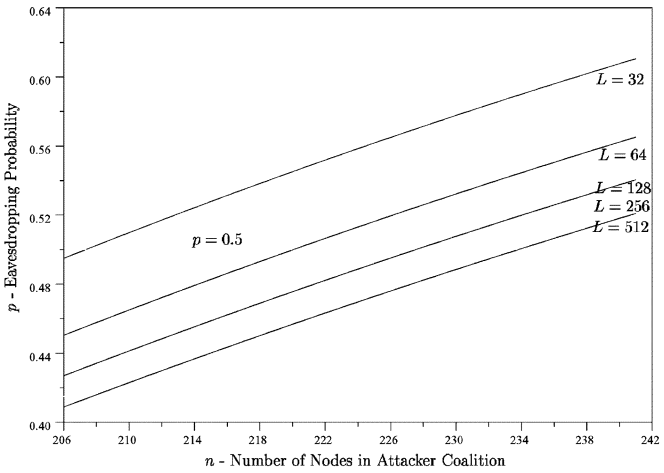


Fig. 5. Plots of eavesdropping probability versus n for various values of L. For all plots $P = 19390$, $k = 1610$.

The justification for fixing $L = 64$ can be seen from the plot of the p versus n for $P = 19390, k = 1610$ for various values of L ranging from 32 to 512 in Fig. 5. As expected, the eavesdropping probability p reduces as L increases, but the improvement is only marginal. For instance, for HARPS with $P = 19390, k = 1610, L = 64$, the eavesdropping probability $p = 0.5$ when $n = 220$ (220 nodes are compromised). Increasing L to 512 provides only a marginal improvement (to $n = 233$ for $p = 0.5$). The small improvement perhaps does not warrant the increase in computational complexity that would result by increasing L.

Outage Probability: For the same parameters $(P = 19390, k = 1610, L = 64)$, and $g = 2$, the outage probability $p_o \approx 5.5 \times 10^{-59}$ for HARPS ("bit-security" of roughly 200 bits), and $p_o \approx 7.4 \times 10^{-54}$ for RPS with $P = 53840, k = 2565$ (bit-security of 180 bits). In other words, outage probability is definitely not an issue in practice!

V. Other Properties of HARPS

Apart from the obvious advantages of HARPS (needs the least value of k and yet deteriorates slower) over other random KPD schemes (RPS and LM), HARPS has numerous *other* advantages. In this section, we shall briefly investigate the "other" properties of HARPS that make it a good candidate for securing ad hoc networks.

A. Multicast Security

Though multicast communication between g nodes can be achieved by $g - 1$ unicast transmissions, multicasting has two obvious advantages—efficient bandwidth usage, and authentication of multicast.[10] Another potential application of multicasting is as a secure replacement for broadcast (by nodes to their neighbors). Broadcast is usually employed for nodes to exchange data for constructing global, dynamic, routing tables. Usually, broadcasts do not need secrecy (they need not be encrypted). However, they need to be *authenticated* to ensure that unauthorized nodes do not make misleading broadcasts.

[10]In some cases, it may be necessary for *each* of the g nodes to know that *all* the g nodes received the message.

Furthermore, for many MANET applications, a major concern is their susceptibility to jamming. Susceptibility to jamming can be drastically reduced if nodes participating in message exchanges share a key, which can be used for code-division multiple-access (CDMA) or frequency hopping (FH). For nodes equipped with some form of KPD, the nodes need to know only the respective IDs before they can establish a key. Very little exchange needs to be done in an "open" channel[11] (the IDs have to be exchanged in an open channel before the shared secret can be established). In practice, nodes could "tune" into an open channel for a very small fraction of time for the purpose of "welcoming" new neighbors. For a large fraction of time, the exchanges between nodes would be relatively immune to jamming.

For securing the physical layer for broadcasts, however, a system wide secret may be needed, which is not a good idea. Replacing broadcasts to neighboring nodes by multicasting (with a shared group key) would alleviate this problem, while also simultaneously providing authentication. If multicasting is used primarily for securing the physical layer (multicast secret used as CDMA/FH key), we could live with much less security for the multicast group secret. The secret is unique only to that group (even if one member is changed the secret will change) and, thus, compromise of the secret would not have any global effect.

Although in theory all KPD schemes are extensible to multicast or conference communications, not all *deployments* are. For instance, a deployment of Blom's scheme for a specific g, n, cannot be used[12] for $g_1 > g$. On the other hand, the probabilistic schemes like LM, RPS, and HARPS can be extended to operate in a multicast scenario (without changing any of the system/deployed secrets) albeit with a reduced margin of security (higher p). Note that our treatment of probability of eavesdropping for HARPS in Section IV was for any g, even though optimizations and subsequent numerical results were reported only for $g = 2$ (or pairwise communications). In general, for pairwise communications we saw that the optimal choice of ξ was proportional to $1/n$. However, for $g > 2$, typically, the optimal choice of ξ would be very close to 1.

The reduced security of multicast group secrets, however, is not a major disadvantage if the multicast secrets are used just for securing the physical layer. Further, for the deterministic schemes, the number of preloaded secrets increase drastically with g (for example, $k = \binom{n+g-1}{g-1}$ for Blom's scheme) and no tradeoff of security (p) versus k is possible (as p can assume only values $0/1$). However, for the probabilistic schemes g could be made larger by sacrificing some security (increasing p). Thus, deployments of probabilistic schemes for MANETS could be optimized for $g = 2$ (unicast communication) and some value of n, and used for multicasts with significantly lower security. For large g, the "weakly" authenticated multicasts could be followed up with individual authentication

[11]Compare this with the situation where PKI is used instead—nodes need to exchange *signed public keys*—or the order of a few thousand bits—before they can establish a shared secret.

[12]This would involve changing the g symmetric polynomial to a g_1 symmetric polynomial, $g_1 > g$ and, therefore, changing of *all* preloaded secrets.

to each node using the much more secure pairwise keys, or authentication provided to smaller subgroups depending on the topology of the network. For example, if all nodes are aware of their immediate and two-hop neighbors, authentication needs to be sent only to a minimum number of nodes which have a path, not passing through the broadcasting node, to all the nodes receiving the broadcast [3].

1) Broadcast Encryption: HARPS also facilitates broadcast encryption [19]. Unlike most broadcast encryption schemes proposed in literature [19]–[21], HARPS facilitates broadcast encryption by the TA and by peers. Consider a scenario, where a node which M neighbors, wishes to establish a shared secret with g of them, which the remaining w nodes ($g + w = M$) should *not* have access to. If $g \ll M$, then the shared secret could be established as a group secret based on the preloaded secrets shared by the g nodes. However, if g is large (or if $w \ll M$) the shared secret could be established through broadcast encryption. The source nodes chooses a secret and encrypts the secret with m keys $B_1 \dots B_m$. The m keys used are chosen from the k preloaded keys that the source node possesses, by excluding all the keys present in the w nodes. For example, the key with index i can be used for this purpose if the source node has the ith key, and if the w nodes together do not have the ith key or if the hash depth of the ith key with the source node is lower than the minimum hash depth among all instances of the ith key the w nodes possess.

Broadcast encryption by the TA could serve another very useful purpose. Nodes suspected of being compromised could be *revoked* from the system by the TA, by broadcasting a secret to all nodes, while selectively excluding the nodes to be revoked. The nodes could periodically check for broadcast messages from the TA (which may be posted at select Web sites). Nodes that do not have access to the Internet could receive the broadcast from other nodes.

B. Resistance to Node Synthesis

The primary difference between conventional key distribution schemes (like Kerberos and PKI) and KPDs is that in the former, revealing secrets in some n nodes $A_1 \dots A_n$ does not provide *any* information about the secrets in an other node A_0, which is obviously not true for KPD schemes. So it is of interest to see how many nodes $A_1 \dots A_n$ an attacker needs to compromise in order to expose *all* secrets buried in node A_0 (not just secrets that could permit an attacker to "eavesdrop"). An attacker who has managed to expose all secrets buried in node A_0 in this fashion has effectively managed to "synthesize" node A_0, using *other* nodes.

For an n-secure deterministic KPD-like Blom's scheme, compromising $n + 1$ nodes is enough to *completely* compromise the system—after which the attacker would be able to, at will, determine *all* keys in *any* node. However, the situation is different for random KPD schemes. The probability that an attacker can synthesize a particular node by tampering with n other nodes is

$$p_s = (1 - \varepsilon_s)^P \tag{14}$$

where

$$\varepsilon_s = \xi \sum_{u=0}^{n} B_\xi(n,u) \frac{1}{L} \sum_{l=1}^{L} \left(\frac{L-l}{L} \right)^u. \tag{15}$$

In order for a key i to be safe, the target node A_0 should have the key i, and in the event u of n attacker nodes have the key i, their minimum hash depth should be greater than the hash depth of the key i in the target node.

For RPS with $P = 53\,840, k = 2565$, the attacker needs $n = 169$—or the attacker has to compromise 169 nodes $A_1 \dots A_{169}$ to have a "reasonable shot" ($p_s = 0.5$) at synthesizing the node A_0. For HARPS with $P = 19\,390, k = 1610$, and $L = 64$, the attacker needs to compromise about 2800 nodes in order to achieve this! For HARPS with the same P, k but with $L = 512$, the attacker needs to compromise over 10 000 nodes! Note that while increasing values of L beyond 64 did not help in reducing the eavesdropping probability p, it helps substantially in reducing p_s, the probability of node synthesis.

Obviously, probability of node synthesis can be reduced further by choosing smaller and smaller values of ξ (which would, however, increase the eavesdropping probability for smaller n). Also, note that as earlier, the comparison is not fair for HARPS as the comparison is between HARPS with $k = 1610$ and RPS with $k = 2565$. Furthermore, RPS also uses a lower value of ξ. In spite of this, HARPS clearly demonstrates a *dramatically higher resistance* to node synthesis than RPS. This feature, as we shall see very soon, is very useful for realization of a long-lived security infrastructure using HARPS.

C. Hierarchical Deployments

The possibility of hierarchical deployments is a significant advantage for practical deployments, both in terms of ease of administration of the system, and localizing security breaches for damage control. While Blom's scheme can be deployed in a hierarchical fashion, it would be at the expense of substantial increase in complexity. For example, for a two-level hierarchy of domains and users the system polynomial would be four variables instead of two variables (for $g = 2$).

The LM scheme offers a simple vertical hierarchy with no additional complexity. The vertical hierarchy is made feasible [4] by selecting nonoverlapping intervals of the hash depth for different levels. For example, for a m level hierarchy, with a hash depth of L for every level, the hash depth of the m different levels would be $1 : L, L+1 : 2L \dots (m-1)L+1 : mL$. The highest (or most "trusted") level would have the least hash depths. Even if keys at a lower level at completely compromised, it will not affect the security of the higher levels (lower hash depths).

HARPS and RPS offer a richer tree structured hierarchy. At the top most level (or level H) is the TA with P keys. Nodes at level $H-1$ have $P_{H-1} \leq P$ keys each, picked from the P keys of the TA. Each parent could act as a TA of its child nodes. The nodes at level $H-2$ would likewise have P_{H-2} keys, which are picked from the pool of P_{H-1} keys belonging to *their* parent. HARPS provides "separation" of the lower levels from parents, as we can choose different ranges of hash depths for each level, like the LM scheme. So, child nodes pick a subset of

the root keys of the parent, and have higher hash depths. Nodes in the same level will have the same hash depth range [22]. Thus, HARPS combines the advantages of both RPS (tree hierarchy) and LM (separation of levels).

D. System Renewal

As the security of all KPDs rely on limiting the number of compromised nodes, usually some form of tamper-resistant or read-proof hardware is called for. However, in spite of efforts made to keep nodes read-proof, secrets are bound to be exposed. As keys get compromised, they need to be constantly renewed to render the exposed keys useless. Updating the keys could be performed by interacting with the TA (for hierarchical deployments the parent, or any node above the parent, can act as a TA). For interaction between TA and a node *all*, the preloaded keys could be used to derive the session key. Note that in practice, it may be infeasible to perform key updates by physical contact with the parent node—so the updates have to be bootstrapped on current secrets, but if an attacker has compromised *all* secrets buried in a node, then the attacker could also participate in key updates. So how does renewal help?

Renewal helps if improvement in technology is able to guarantee even "partial" read-proof ability [4] (a guarantee that an attacker, however sophisticated, under constraints of finite time (period between two updates), may not be able to expose *all* bits buried in a node by tampering with it). Under such a guarantee, even if the attacker is able to expose partial secrets from each node, he is forced to tamper with *other* nodes to expose the remaining secrets buried in a target node. As we have already seen, HARPS has a very high resistance to node synthesis and, thus, may be extremely impractical for an attacker to carry out such large scale attacks within the duration between two updates. Under such an assumption, an attacker may need to tamper with a few hundred thousand nodes (see [23]) before he can effectively synthesize a node that can take part in key updates.

One of the main advantages of hierarchical deployments come from the flexibility it offers for updates. Updates are handled by many parents rather than by a single TA. Only the level $H-1$ nodes directly below the TA need to contact the TA for updates. This could go a long way in reducing "update floods." Nevertheless, it still might not be practical for performing renewal of all nodes synchronously. Thus, there will be situations where a node with its keys updated may need to communicate with another node, which has not had the opportunity to do so. In order to make this possible, the KPD should permit partial renewal of secrets—which could be done for all random KPD schemes. On the other hand, for deterministic KPD schemes, even if a subset of the TA's secrets are changed, *all* preloaded secrets will need to be changed. This makes random KPD schemes particularly attractive for system renewal. HARPS and LM, however, offer another possibility—the updated secrets could be pre-images of old secrets—under a cryptographically strong one-way function. This would, however, need a long one-way hash chain [24], of the secrets to be created before deployment. Under this circumstance, it is trivial to ensure that nodes with updated keys could still communicate with nodes that have not updated their secrets.

Thus, three very useful properties of HARPS—tree hierarchy, renewal using pre-images, and its tremendous resistance to node synthesis—each in their own way, contribute to making HARPS a very attractive choice for periodic renewal of keys, which is needed for any long-lived deployment that relies on tamper resistant/read-proof hardware. Additionally, the fact that HARPS support broadcast encryption can be used to revoke nodes suspected of being compromised between renewals. Note that while renewal needs a direct access to the TA (or parent node), broadcast encryption does not, and can thus be realized very efficiently. Nodes revoked by broadcast encryption would not be allowed to take part in further renewals, thus making the revocation permanent.

E. Broadcast Authentication

The purpose of broadcast authentication is to provide a means for nodes to verify the authenticity of the source of a message. Broadcast authentication with preloaded secrets can be performed by appending the message with k key-based message authentication codes (MACs), corresponding to each shared secret the source node has [25]. Broadcasts by the TA would be appended with P MACs. Any node, with a very high probability, will be able to verify some of the k MACs. We shall refer to the appended MACs as the *signature* of the source.

While the principle behind broadcast authentication is very similar for RPS and HARPS, the advantage of HARPS is that of the ability of the source node to *choose* the hash depth of the key it uses for the MACs [26]. It could choose any depth as long as it is not smaller than the hash depth the source node has for that key. This ability could be used to strengthen the security of the signature (reduce the probability that an attacker who has compromised n nodes could *forge* the signature) compared with RPS.

Conventional broadcast authentication schemes do not target "intended recipients,"—or any node is equally capable of authenticating a broadcast. In other words, an attacker whose intent is to "fool" a node into accepting the authenticity of a fake message, faces the same amount difficulty in fooling any node. However, in practice, most messages do in fact have "intended" recipients. With HARPS, the ability of the source node to *choose the hash depth* results in some interesting possibilities. This ability could be used to *strengthen* the signature such that it is much more difficult for an attacker to fool select (intended) recipients than other nodes (which may not be of consequence anyway).

VI. Comparison With MBS Scheme

Another "improvement" to RPS scheme has been independently proposed in [5] and [6], where the authors combine RPS with Blom's [8] KPD. In these schemes, the TA chooses P_b independent instances of n_b-secure Blom's KDS. Each node receives secrets corresponding to k_b of those schemes, where $k_b \leq P_b$. Each instance of n_b-secure Blom's scheme needs n_b+1 keys to be preloaded in each node—or a total of $k = k_b(n_b+1)$ keys are preloaded in each node. We shall refer to these schemes as multiple Blom's scheme (MBS).

Let us assume that two nodes with k_b Blom's KPDs each share m of them. In order to eavesdrop on the communication, an attacker needs to expose $n_b + 1$ instances of each shared scheme. Equivalently, MBS schemes can be considered as RPS with $P = P_b$ and $k = k_b$, where the attacker coalition needs $n_b + 1$ "instances" of each shared key in order to obtain the shared secret (for RPS, we need to get only one instance of each shared key).

Let $\xi = (k_b)/(P_b)$. The probability that the ith Blom scheme is not compromised is then

$$\epsilon = \xi^2 \sum_{u=0}^{n_b} B_\xi(n, u) \tag{16}$$

and the probability of eavesdropping is $p = (1 - \epsilon)^{P_b}$.

Just as HARPS is a generalization of RPS and LM, MBS can be considered as a generalization of RPS and Blom's scheme. MBS with $n_b = 0$ reduces to RPS, and MBS with $P_b = k_b = 1$ reduces to Blom's scheme. In general, for small n_b, MBS is more "RPS-like" and for large n_b, more "Blom-like." The tradeoffs here take the form of storage efficiency (reducing number of preloaded keys in each node) versus gracefulness of degradation. For instance, a 20-secure Blom's scheme needs only 21 keys preloaded in each node. However, for the same case, RPS needs more than 2600 keys in each node (to achieve $p < 10^{-20}$ for $n < 20$). However, RPS, unlike Blom's scheme, does not fail catastrophically. What MBS tries to do is improve the efficiency of RPS, but however, as we shall see very soon, at some loss. MBS *deteriorates faster* than RPS with increasing n. For the sake of comparison of MBS with RPS and HARPS, we shall consider design of KPD's optimized for $n^* = 20$ (to resist 20 compromised nodes). For all schemes, we shall use the same number (1650) of preloaded keys.

For HARPS, the optimal choice of P is roughly 20 000. For RPS, the optimal choice is roughly $P = 35\,000$. For MBS, we shall consider[13] three cases—$n_b = 2, 4$, and 10. For $n_b = 2, k_b = (1650)/(n_b + 1) = 550$. The optimal choice of P_b [which maximizes ϵ in (16)] is $P_b = 5000$. The optimal choices for the other two cases are $P_b = 1940$ $(n_b = 4, k_b = 330)$ and $P_b = 363$ $(n_b = 10, k_b = 150)$. Fig. 6 depicts plots of probability of eavesdropping (y axis chosen as $-\log(p)$) versus n, the number of compromised nodes for RPS, HARPS and the three cases for MBS. Note that MBS with $n_b = 10$ performs significantly better than RPS and even better than HARPS for $n = 20$. However, note that the performance deteriorates very fast as n increases. For other cases, MBS still performs better than RPS at $n = 20$ but as expected, deteriorates faster than RPS, but slower than the case where $n_b = 10$ (the most "Blom-like" of the three cases). HARPS on the other hand, as we have already seen, performs better than RPS for *all* n. The degradation of security for HARPS is significantly slower than that of RPS. In addition, MBS also has carries the other inherent disadvantages of Blom's scheme, like complexity of computations for determining shared secrets (evaluating polynomials in a prime field) and complexity of hierarchical extensions.

[13]The choice of $k = 1650$ is due to the fact that 1650 has 3, 5, and 11 as factors.

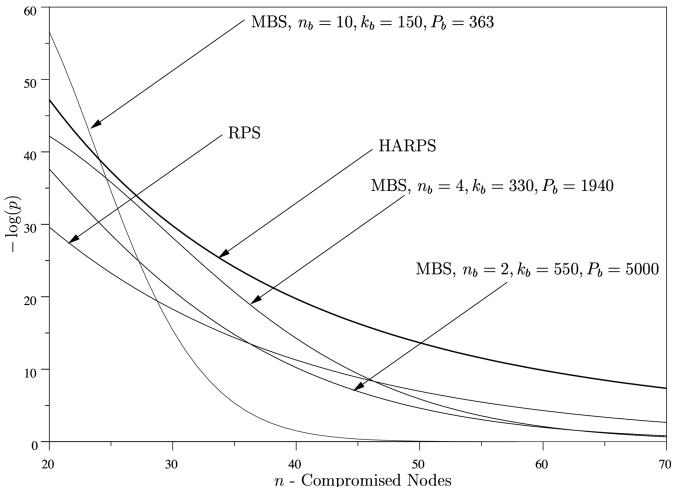


Fig. 6. Comparison of RPS, HARPS, and MBS (for $n_b = 2, 4, 10$).

VII. Conclusion

This paper proposed HARPS, a novel and efficient probabilistic key predistribution distribution scheme, and an analysis of it's merits. The paper also addressed many desirable properties that a key distribution system should possess to aid practical deployments, like the possibility of hierarchical deployments, extensions to multicast scenarios, ease of post deployment extensions, key renewal, mechanisms for nonrepudiation and revocation of nodes.

While KPD schemes enable simple and effective means of building trust relationships, their biggest disadvantage is their vulnerability to the ability of an adversary to tamper with and expose hidden secrets from many nodes. That the nodes need not be tamper resistant, but just need the ability to reset all keys when tampered with [1], or be relatively immune to passive "sniffing" of keys, perhaps is not a very unreasonable assumption. Further, if advances in technology is able to provide *partial* read-proof guarantees, it is still possible to renew the KPD by bootstrapping on old secrets, resulting in a long-lived KPD.

It was shown that the probabilistic KPD's, in general, offer more flexibility for securing deployments of resource constrained AHN's. Among the probabilistic methods, HARPS, a generalization of RPS and LM was shown to have significantly better performance, and deployment flexibility. HARPS, was shown to be highly storage efficient compared to other random KPD schemes, while at the same time offering slower deterioration of security with increasing numbers of compromised nodes. Further, four important properties of HARPS: 1) ability to deploy in a tree-hierarchical fashion; 2) key renewal using pre-images; 3) high resistance to node synthesis; and 4) ability to achieve revocation through broadcast encryption—contribute in their own way in making HARPS a very desirable KPD scheme, especially for long-lived deployments of AHNs. That HARPS permits broadcast authentication is useful to achieve nonrepudiation. Additionally, HARPS also offers a novel, and potentially useful cryptographic application of "targeted" signatures.

While many KPD schemes have been proposed in literature which exclusively target establishment of shared secrets

[3]–[18] or broadcast authentication [25] or broadcast encryption [19]–[21], HARPS caters for *all* three applications with the *same set of preloaded keys*. These factors, together with the unique features of HARPS that aid key renewal, makes HARPS particularly well suited as an enabler for a security infrastructure for resource constrained environments [27].

REFERENCES

[1] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin, "Tamper proof security: Theoretical foundations for security against hardware tampering," in *Conf. Theory Cryptography*, Cambridge, MA, Feb. 2004.

[2] J. Lotspiech, S. Nusser, and F. Pestoni, "Anonymous trust: Digital rights management using broadcast encryption," *Proc. IEEE*, vol. 92, no. 6, pp. 898–909, 2004.

[3] M. Ramkumar, N. Memon, and R. Simha, "Preloaded key based multicast and broadcast authentication in mobile ad hoc networks," *Proc. GLOBECOM*, pp. 1405–1409, Dec. 2003.

[4] T. Leighton and S. Micali, "Secret-key agreement without public-key cryptography," in *Advances in Cryptology—CRYPTO*, 1994, pp. 456–479.

[5] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key predistribution scheme for wireless sensor networks," in *Proc. 10th ACM Conf. Comput. Commun. Security*, 2003, pp. 42–51.

[6] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," presented at the 10th ACM Conf. Comput. Commun. Security, Washington, DC, 2003.

[7] T. Matsumoto and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Dec. 1976.

[8] R. Blom, "An optimal class of symmetric key generation systems," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 1984, vol. 209, Proc. of Eurocrypt, Advances in Cryptology, pp. 335–338.

[9] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," *Lecture Notes in Computer Science*, vol. 740, pp. 471–486, 1993.

[10] L. Gong and D. J. Wheeler, "A matrix key distribution scheme," *J. Cryptology*, vol. 2, no. 2, pp. 51–59, 1990.

[11] C. J. Mitchell and F. C. Piper, "Key storage in secure networks," *Discrete Appl. Math.*, vol. 21, pp. 215–228, 1995.

[12] M. Dyer, T. Fenner, A. Frieze, and A. Thomason, "On key storage in secure networks," *J. Cryptology*, vol. 8, pp. 189–200, 1995.

[13] C. Padro, I. Gracia, S. Martin, and P. Morillo, "Linear broadcast encryption schemes," *Discrete Appl. Math.*, vol. 128, no. 1, pp. 223–238, 2003.

[14] P. Erdos, P. Frankl, and Z. Furedi, "Families of finite sets in which no set is covered by the union of r others," *Israel J. Math.*, vol. 51, pp. 79–89, 1985.

[15] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. 9th ACM Conf. Comput. Commun. Security*, Washington, DC, Nov. 2002, pp. 41–47.

[16] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," presented at the IEEE Symp. Security Privacy, Berkeley, CA, May 2003.

[17] R. Di Pietro, L. V. Mancini, and A. Mei, "Random key assignment for secure wireless sensor networks," in *Proc. ACM Workshop Security Ad Hoc Sensor Networks*, Oct. 2003.

[18] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach," presented at the 11th IEEE Int. Conf. Netw. Protocols (ICNP'03), Atlanta, GA, Nov. 4–7, 2003.

[19] A. Fiat and M. Noar, "Broadcast encryption," in *Lecture Notes Computer Science, Advances in Cryptology*. New York: Springer-Verlag, 1994, vol. 773, pp. 480–491.

[20] C. K. Wong, M. Gouda, and S. Lam, "Secure group communications using key graphs," in *Proceedings of SIGCOMM 98*, 1998, pp. 68–79.

[21] D. Noar, M. Noar, and J. Lotspiech, "Revocation and tracing routines for stateless receivers," in *Lecture Notes in Computer Science, Advances in Cryptology*: Springer-Verlag, 2001, vol. 2139.

[22] M. Ramkumar, K. S. Ayeegoundanpalayam, N. Memon, and R. Simha, "A hierarchical random key predistribution scheme for ad hoc networks of pervasive devices," in *12th Annu. Netw. Distrib. Syst. Security Symp. (NDSS)*, 2005, submitted for publication.

[23] M. Ramkumar and N. Memon, "On the security of random key predistribution schemes," in *Proc. 5th IEEE SMC-IA Workshop*, West Point, NY, Jun. 2004, pp. 153–160.

[24] L. Lamport, "Password authentication with insecure communication," in *Commun. ACM*, vol. 24, Nov. 1981, pp. 770–772.

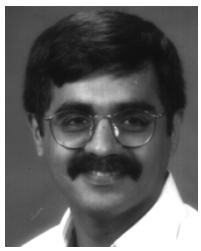
[25] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas, "Multicast security: A taxonomy and some efficient constructions," in *Proc. INFOCOMM*, New York, Mar. 1999, pp. 708–716.

[26] M. Ramkumar and K. S. Ayeegoundanpalayam, "An efficient source authentication scheme with hashed preloaded subsets," Manuscript under preparation.

[27] M. Ramkumar and N. Memon, "KPI—An alternative to PKI for pervasive networks," *IEEE J. Sel. Areas Commun.*, submitted for publication.

Mahalingam Ramkumar received the B.S. degree from the University of Madras, Madras, India, in 1987, the M.S. degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India, in 1997, and the Ph.D. degree in electrical engineering from the New Jersey Institute of Technology, Newark, NJ, in 2000.

He is an Assistant Professor in the Department of Computer Science and Engineering, Mississippi State University, Mississippi State, MS. His research interests include network security, cryptography, ad hoc networks, data hiding, and multimedia security.



Nasir Memon received the B.E. degree in chemical engineering and the M.Sc. degree in mathematics from the Birla Institute of Technology, Pilani, India, and the M.S. and Ph.D. degrees in computer science from the University of Nebraska, Lincoln.

He is a Professor in the Computer Science Department at Polytechnic University, Brooklyn, NY. His research interests include data compression, computer and network security, multimedia data security, and multimedia communications.