I-HARPS: An Efficient Key Predistribution Scheme for Mobile Computing Applications

Mahalingam Ramkumar
Department of Computer Science and Engineering
Mississippi State University, Mississippi State, MS 39762.
Email: ramkumar@cse.msstate.edu.

***Abstract*— With rapidly decreasing cost of storage, even for mobile computing applications involving PDAs / mobile phones storage (using flash memory) is an *inexpensive resource*. We introduce a novel probabilistic key predistribution scheme (PKPS) I-HARPS which can make good use of this inexpensive resource to improve security. I-HARPS is a combination of random subset allocation schemes first proposed by Dyer et al [1] in 1995, and the escrowed master key based key distribution scheme proposed by Leighton and Micali in 1993 [2]. While PKPSs have received substantial attention recently in the context of highly resource constrained sensor networks, we argue that the fact that I-HARPS can resist coalitions of even *millions* of nodes with very low computational complexity, and very reasonable storage requirements, can significantly expand the scope of applications of PKPSs.**

I. Introduction

In many emerging application scenarios devices are expected to co-operate with each other to realize synergistic benefits. For example, in MANETs [3] fixed and mobile wireless devices will exchange information to facilitate co-operative routing. In such application scenarios, *cryptographic authentication* of data disseminated by nodes, is essential for reducing the ill-effects of malicious nodes that could disseminate misleading information.

Cryptographic authentication is facilitated by *security associations* (SA), which in turn are facilitated by key distribution schemes (KDS). A key distribution scheme (KDS), typically consisting of a key distribution center (KDC) and N nodes (N is the network size) is a mechanism for distributing secrets to all nodes in the network to facilitate establishment of cryptographic "bonds" or SAs between the nodes. While many types of SAs are possible, in this paper we limit ourselves to mutual authentication through establishment of pair-wise shared secrets.

The use of SAs for securing interactions imposes some costs, due to the need for bandwidth, computational and storage overheads. While the costs associated with all resources are rapidly dropping (Moore's law), for mobile computing applications bandwidth and computational ability are still constrained by the need to prolong battery life of mobile devices. Thus storage is arguably the cheapest of resources - especially considering the fact that flash based storage cards up to 8 GB are already very common. Using say a few tens or even hundreds of MBs of storage for enhancing the security may be acceptable, if it can reduce reliance on more expensive resources like bandwidth and computational complexity.

In wireless applications, it is often desirable for KDSs to facilitate *ad hoc* authentication (without the need for a trusted third party). While certificates based public key schemes facilitate ad hoc authentication, they have high computational overheads, and large bandwidth overheads for disseminating certificates. ID based public key schemes [4] have garnered significant interest recently as they facilitate ad hoc authentication without the need for disseminating certificates. However such schemes can be even more computationally expensive than certificates based public key schemes.

Key predistribution schemes (KPS) [5] are ID-based (facilitate ad hoc SAs without disseminating certificates), and use only inexpensive symmetric primitives. However, they are susceptible to *collusions*. Increasing collusion resistance of *most* KPSs call for 1) increasing storage for secrets and 2) increased *computational complexity*. While increase in storage requirement is not a serious issue, increase in computational complexity is *not* acceptable.

The contribution of this paper is a novel KPS, I-HARPS, for which the *collusion resistance is limited only by storage*. Furthermore, the storage does not need *any* protection. By taking advantage of the large amounts storage that are readily available to hand-held devices, I-HARPS can even resist collusions of many hundreds of thousands of nodes, with *very low* computational complexity.

The rest of this paper is organized as follows. In Section II we briefly review KPSs with more focus on probabilistic KPSs. We review the PKPS based on random subset allocation first suggested by Dyer et al [1] in 1995, and the escrowed master key based KDS proposed by Leighton and Micali [2] in 1993, which serve as the foundations for the proposed KDS discussed in Section III. Conclusions are offered in Section IV.

II. Key Predistribution Schemes

Key predistribution schemes consist of a key distribution center (KDC), and a set of N nodes with *unique* IDs. The KDC chooses a set of P independent secrets $\mathbb{S} = \{K_1 \cdots K_P\}$. Each node is provided with k secrets. The set of secrets provided to any node is a function of the P secrets chosen by the KDC, and the unique ID of the node. Typically, there is *very little constraint* on the size P. For most KPSs the KDC can derive all P secrets using just one (or a few) secrets, and a

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

strong one-way function[1] $h()$. However the number of secrets *assigned* to each node, k, will have some constraints.

A. Nonscalable and Scalable KPSs

KPSs can be classified into two basic categories - scalable (KPS) and nonscalable (NKPS). A trivial example of a non-scalable KPS is the "basic" key distribution scheme, where the KDC chooses $P = \binom{N}{2}$ secrets, and assigns $k = N - 1$ secrets to each node.

A less trivial example of an NKPS is the KDS proposed by Leighton and Micali [2] (LM-KDS), in which the KDC chooses a master key K and a hash function $h()$, and provides node A with the secret $K_A = h(K, A)$. To facilitate mutual authentication of nodes A and B, node A is provided with a public value

$$P_{AB} = h(K_A, B) \oplus h(K_B, A) \tag{1}$$

Only nodes A and B can determine $K_{AB} = h(K_B, A)$. Node B by direct evaluation of $K_{AB} = h(K_B, A)$, and node A by evaluating

$$h(K_A, B) \oplus P_{AB} = h(K_B, A) = K_{AB} \tag{2}$$

Note that the total number of possible public values are $\binom{N}{2}$, and any node may need upto $N - 1$ of those values. For *very large* network sizes storing $N - 1$ public values may not be feasible.

1) Scalable KPSs: Scalable KPSs (or just KPSs) are inherently trade-offs between security and complexity. That such trade-offs are possible was first realized by Blom et al [5] who proposed the first KPS in the literature. KPSs themselves may be divided into two broad categories - deterministic and probabilistic KPSs.

Most KPSs based on finite field arithmetic belong to the former category. For example, in a n-secure Blom's scheme [5] with a network size of N, the KDC chooses $P = \binom{k}{2}$ secrets from $\mathbb{Z}_q = \{0, 1, \ldots, q-1\}$, where $n = \lceil \frac{k+1}{2} \rceil$ and $q \geq N$ is a prime, and generates a symmetric polynomial $f(x, y)$, of order $k - 1$. The P coefficients of $f(x, y)$ are the secrets chosen by the KDC. Every node is assigned a *unique* ID from $\mathbb{Z}_q$. A node A (node with ID $A \in \mathbb{Z}_q$) receives $g_A(x) = f(x, A)$ securely - or node A receives the k coefficients of $g_A(x)$ securely from the KDC. Now two nodes A and B can calculate $K_{AB} = K_{BA} = f(A, B) = f(B, A) = g_A(B) = g_B(A)$ independently.

By extracting k secrets each from *more* than $n = \lceil \frac{k+1}{2} \rceil$ nodes an attacker can construct a system of P independent simultaneous equations to solve for all P secrets chosen by the KDC. Even with $P - 1$ simultaneous equations however, the attacker learns nothing. A n-secure Blom's KPS is thus unconditionally secure as long as n or less nodes have been compromised, and is completely compromised if more than n nodes are compromised - or the failure occurs catastrophically.

[1]For example a master secret M could be used to derive all $K_i = h(M \parallel i)$, $1 \leq i \leq P$. Only the secret M needs to be stored, and all other secrets generated on-demand.

2) Scalability vs Collusion Resistance: With $\mathbb{O}(n)$ limitation on k, NKPSs can only support network sizes of $N = \mathbb{O}(n)$. Scalable KPSs on the other hand, can support *any* network size N (N is limited only by the number of bits chosen to represent the ID of each node as each node requires a unique ID), but can only *tolerate* collusions of n or less nodes.

B. Probabilistic KPS

The concept of n-secureness of a scalable KPS is however not an adequate description of *probabilistic* key pre-distribution schemes (PKPS). For any PKPS, by exposing secrets from n nodes an attacker can discover shared secrets between arbitrary nodes (not part of the n compromised nodes) with a some probability p. Thus PKPSs are more aptly characterized as (n, p)-secure. Note that there is nothing objectionable about "probabilistic assurances." Even for a "strong" symmetric cipher employing a 128-bit key, there is always a *probability* that the attacker can *guess* the key in a single attempt. For 128-bit keys the probability that an attacker can "pull the secret out of a hat" is roughly $p \approx 10^{-39}$. Thus as long as the probability of failure p for PKPSs is small, probabilistic assurances are in no way inferior.

Most probabilistic KPSs are based on the idea of random allocation of subsets of keys to each node, from a pool of keys chosen by the KDC. Such PKPSs are extensions of similar earlier techniques [6] - [8] that relied on *deterministic* strategies for allocation of subsets of keys to every node. Dyer et al [1] were was the first to point out the simplicity and effectiveness of *random subset allocations*. Dyer et al also provided an elegant analysis of the security of such schemes. Dyer's work on random subset allocations has also been used for broadcast authentication [9]. More recently, this idea has received substantial attention in the context of sensor networks [10] - [12]. In this paper, we refer to all such schemes [1], [10] - [12] as RPS (random preloaded subsets).

1) RPS: RPS [12] is determined by two parameters P and k. The KDC chooses a indexed set of P secrets $\mathbb{S} = \{K_1, K_2, \ldots, K_P\}$. For a maximum network size of N, every node is assigned a unique d-bit ID, where $d > \log_2 N$. Every node is also assigned $k = \xi P$ secrets which are randomly chosen subsets of $\mathbb{S}$. A public function $F()$ determines the indexes of secrets assigned to any node. Thus a node with ID A is assigned indexes $F(A) = \{A_1, A_2, \ldots, A_k\}$ and therefore the corresponding secrets $\mathbb{S}_A = \{K_{A_1}, K_{A_2}, \ldots, K_{A_k}\}$.

Any two nodes will share $\bar{m} = \xi k$ such keys on an average. Specifically, two nodes A and B will share $m \approx \bar{m}$ indexes determined by $F(A) \cap F(B) = \{I_1 \cdots I_m\}$. In other words, the secrets shared by A and B are $\{K_{I_1} \cdots K_{I_m}\}$. Thus A and B can independently calculate their SA K_{AB} as a secure one-way function of *all* m shared secrets (for example hashing all secrets together using a cryptographic hash function $h()$). Note that an attacker has to discover *all* the m secrets that A and B share, to discover K_{AB}.

The probability p with which an attacker who has exposed all secrets from n nodes, can discover all shared secrets

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

between two nodes (which are not a part of the n compromised nodes) is [1], [13]

$$p(n) = (1 - \epsilon(n))^k \text{ where } \epsilon(n) = \xi(1-\xi)^n. \quad (3)$$

The rationale for the equation above can be seen by considering a k round game between the node B, who can "legally" discover K_{AB}, on one side, and an n attacker nodes (who try to determine K_{AB} through "objectionable" means) on the other side. In the first round of the game all $n+1$ players (B and the n attackers) attempt to pick the secret with index A_1 - or K_{A_1}, and succeed with probability $\xi = k/P$ (and fail with probability $1-\xi$). Similarly, in the second round they attempt to pick K_{A_2}, and so on.

Node B winds round i if 1) B picks K_{A_i}, *and* 2) none of the n attackers pick K_{A_i}. The probability ϵ that B wins any round is $\epsilon = \xi(1-\xi)^n$. While the probability that B wins any round is low, all that B has to do to "win the match" is win a single round. Even if the attacker loose a single round A and B will share an elementary secret used for K_{AB} the attackers cannot discover (and so the attackers cannot discover K_{AB}). Thus the probability $p(n)$ that the attacker's can win every round (or discover K_{AB}) is given by Eq (3).

III. RPS WITH ID-HASHING

ID-HAshed RPS (I-HARPS), the novel PKPS proposed in this section, is "a scalable extension" of an NKPS, achieved by utilizing multiple instances of NKPSs in parallel. I-HARPS is defined by three parameters P, ξ and L. Like any KPS, I-HARPS consists of a KDC and N nodes with unique (at least $\log_2 N$-bit) IDs. I-HARPS, which is a generalization of RPS and LM-KDS, consists of P "mini" LM-KDS deployments in parallel. Each mini LM-KDS is however limited to a network size of L.

Each node with a $\log_2 N$-bit ID is assigned many $\log_2 L$-bit "minor-IDs". Specifically, each node is assigned a minor ID in k of the P mini LM-KDSs where $k \approx \bar{k} = \xi P$. Corresponding to each minor ID, nodes are provided with a secret and $L-1$ public values (amounting to a total of k secrets and $k(L-1)$ public values for each node).

In I-HARPS, as in RPS, the KDC chooses an indexed set of P secrets $K_1 \cdots K_P$, which are now the master keys of the P mini LM-KDSs. A public function $F()$ (as in RPS) determines the indexes of k LM-KDS (out of P) assigned to any node. Thus for node A, $F(A) = \{A_1 \cdots A_k\}$. A second public function $a_{A_i} = f(A, A_i), 1 \le i \le k$ is used to generate a uniformly distributed integer between 1 and L, corresponding to each assigned index. Now $\{a_{A_1} \cdots a_{A_k}\}$ are the k minor-IDs of node A in the mini LM-KDS systems indexed by $\{A_1 \cdots A_k\}$. Node A is assigned the set of secrets

$$\mathbb{S}_A = \{K_{A_1}^{a_{A_1}} \cdots K_{A_k}^{a_{A_k}}\}, \quad K_{A_i}^{a_{A_i}} = h(K_i, a_{A_i}), 1 \le i \le k.$$

In addition, node A is also assigned $L-1$ "public values" corresponding to each of the k secrets assigned to A. For instance, the $L-1$ public values corresponding to the index A_i (and the secret $K_{A_i}^{a_{A_i}}$) are

$$P_{A_i}^{a_{A_i}, j} = h(K_{A_i}^{a_{A_i}}, j) \oplus h(K_{A_i}^{j}, a_{A_i}), \quad (4)$$

where j takes all values from the set $\{1, 2, \ldots, a_{A_i}-1, a_{A_i}+1, \ldots L\}$ of cardinality $L-1$.

2) Evaluation of Security Association K_{AB}: Let $\{I_1 \cdots I_m\}$ be the indexes shared by A and B (or $\{I_1 \cdots I_m\} = F(A) \cap F(B)$). Let the corresponding minor IDs of A and B in the m mini LM-KDS systems be $\{a_{I_1} \cdots a_{I_m}\}$ and $\{b_{I_1} \cdots b_{I_m}\}$ respectively. Now define $S_{I_i} = h(K_{I_i}^{b_{I_i}}, a_{I_i}), i = 1 \cdots m$.

The SA K_{AB} for mutual authentication of A and B is now evaluated using m secrets $\{S_{I_1} \cdots S_{I_m}\}$. Both nodes A and B can evaluate all (m) such S_is. While node B (which has access to $K_{I_i}^{b_{I_i}}$) can directly evaluate S_{I_i}, node A cannot do so directly. This the reason for providing node A with the public values, with which A can evaluate any S_{I_i} as

$$S_{I_i} = h(K_{I_i}^{a_{I_i}}, b_{I_i}) \oplus P_{I_i}^{a_{I-i}, b_{I_i}} = h(K_{I_i}^{b_{I_i}}, a_{I_i}). \quad (5)$$

The SA K_{AB} between A and B can now be derived independently by both A and B using all m "elementary shared secrets" $S_{I_1} \cdots S_{I_m}$ as $K_{AB} = h(S_{I_1} \parallel S_{I_2} \parallel \cdots \parallel S_{I_m})$.

A. Analysis

The analysis of I-HARPS can once again be performed by considering a k round game. The difference is that while the probability that B can discover an elementary secret $S_{A_i} = h(K_{A_i}^{b_{A_i}}, a_{A_i})$ (used for deriving K_{AB}) is ξ, where a_{A_i} and b_{A_i} are the "short-IDs" of A and B respectively corresponding to the index A_i, for calculating the elementary secret it is not sufficient for an attacker (say C) to pick the index A_i. Apart from picking the index A_i, the short ID of the attacker c_{A_i} should be such that $c_{A_i} \in \{a_{A_i}, b_{A_i}\}$.

As $a_{A_i}, b_{A_i}, c_{A_i}$ are iid and uniformly distributed integers between 1 and L,

$$\Pr\{c_{A_i} \in \{a_{A_i}, b_{A_i}\}\} = \frac{2L-1}{L^2} = \gamma \approx \frac{2}{L}. \quad (6)$$

Thus the probability that B wins any round is $\epsilon' = \xi(1-\gamma\xi)^n$, and the probability $p'(n)$ that the attacker can discover K_{AB} is $p'(n) = (1-\epsilon')^k$. Note that

$$\epsilon'(n) = \xi(1-\xi\gamma)^n \approx \xi(1-\xi)^{n\gamma} = \epsilon(n\gamma). \quad (7)$$

Thus for the same k and P, the number of compromised nodes that I-HARPS can tolerate is increased by a factor $1/\gamma \approx L/2$. However, while RPS needs storage of k secrets, I-HARPS needs storage for k secrets and $k(L-1)$ public values.

B. PKPS Efficiencies

For RPS, the optimal choice of ξ (which minimizes $\bar{k}$ for some desired upper bound on the probability of attacker success) is $\xi^* = 1/(n+1) \approx 1/n$ for large n. Correspondingly, k, P and $\bar{m}$ are

$$k \approx n\bar{m} \quad P \approx \bar{m}n^2 \quad \bar{m} = \mathrm{e}\log(1/\mathrm{p}) \quad (8)$$

From Eq (7) it is clear that if for some P and ξ, RPS is (n,p)-secure, I-HARPS with P, ξ is $(n/\gamma, p)$-secure (by choosing $L \approx 2/\gamma$). Thus the optimal choice of P and ξ for an (n,p)-secure I-HARPS is the same as that of an $(n\gamma, p)$-secure RPS

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

scheme. Thus for a (n,p)-secure I-HARPS, $\xi^* \approx \frac{1}{\gamma n}$ (subject to the condition that $\xi \leq 1$), and analogous to Eq (8) for RPS, we have

$$k \approx n\gamma\bar{m} < \tfrac{2n\bar{m}}{L} \quad \bar{m} = \mathrm{e}\log(1/\mathrm{p}). \tag{9}$$

and $P \approx \gamma^2 n^2 \bar{m} < \frac{4n^2\bar{m}}{L^2}$. In addition, for I-HARPS we also need to account for storage of the public values. For each of the P keys, the KDC generates $\binom{L}{2} < L^2/2$ public values. Thus the total number of public values to be generated by the KDC is less than $2\bar{m}n^2$. As each node needs to store k secrets and $k(L-1)$ public values, the total storage complexity for each node is $Q \approx 2\bar{m}n$. A comparison of the overheads involved for (n,p)-secure RPS and I-HARPS are summarized together in the table below.

Parameter	RPS	I-HARPS
k (Secret Storage in nodes)	$\bar{m}n$	$\frac{2n}{L}\bar{m}$
Q Total Storage in nodes	$\bar{m}n$	$2\bar{m}n$

The value k determines the computational complexity of the public function, and Q is the storage complexity. It can be readily seen that for (n,p)-security, I-HARPS requires twice as much storage (including storage for public values) as RPS. However computational complexity for $F()$ can be *reduced to any desired extent* by increasing L (as $k \propto n/L$). Thus the security of I-HARPS is limited *only* by available (unprotected) storage. Arguably, in most evolving application scenarios, storage is less constrained than computational ability.

In most scenarios where multiple secrets need to be stored, it is common practice [14] to encrypt all secrets using a single master secret which is afforded a higher level of protection. Or all secrets can be stored encrypted in storage locations that do not need any protection. Thus, to be fair, the fact that I-HARPS needs to store less number of keys may not seem like an advantage. However, note that for scenarios where A and B interact, only *one* node needs access to the public values. So in many application scenarios involving a storage constrained device and another device that is less constrained, this feature can be very useful. For instance node A (which stores secrets and public values) may be a PDA used to query a tiny wireless sensor B (which stores only k secrets). As another example B may be a infra-red / blue-tooth remote control device operating a more capable set top box A.

The main advantage of I-HARPS, however, is that the complexity of the evaluating $F(A) \cap F(B)$ can be *significantly* lower for I-HARPS, as illustrated in the next section with an example.

C. Numerical Illustration

Consider RPS with $P = 2^{25}$ 64-bit keys, and $k = 2^{16}$ (or $\xi = 1/2^9$). For such a scheme an attacker needs to expose all secrets from $n = 512$ devices to discover SAs (like K_{AB}) with probability $p \approx 3.7 \times 10^{-21}$. Alternately, the attacker can expose one in 2.7×10^{20} pair-wise SAs by exposing all secrets from 512 nodes. For $\xi = 2^t$ efficient evaluation of $F(A)$ will involve generation of k uniformly distributed number between 1 and 2^t - or generation of kt random bits. Thus RPS involves generation of $2 \times 9 \times 2^{16}$, or 1.18 million random bits. The storage complexity for $k = 2^16$ is 512 KB for k 64-bit secrets, and roughly $\bar{m} = \xi^2 P = 128$ cryptographic operations with secrets are required. If we desire to increase n by a factor 2^7 (to $n \approx 65000$), k would increase to 2^{16+7}, implying storage of 64 MB for secrets, and complexity of $F(A) \cap F(B)$ of $2 \times 16 \times 2^{23} = 268$ million bits.

On the other hand, consider I-HARPS with $L = 2^{14}$, $k = 2^{10}$, and $P = 2^{13}$. Such a scheme can resist compromise of about 66,000 devices for the same $p \approx 3.7 \times 10^{-21}$. The approach here is to choose the parameters P, ξ for a $(2^3 = 8, 10^{-20})$ RPS and choose $L = 2^{14}$ (or $\gamma \approx 1/2^{13}$) so that $8/\gamma$ is around the desired n (about 2^{16}). Evaluation of $F(A) \cap F(B)$ involves generation of $2 \times 2^{10} \times 3 = 6144$ bits (as $k = 2^{10}$ and $\xi = 1/2^3$). The second public function $f()$ (generating $2m = 256$ 14-bit integers or minor IDs) involves generation of 3584 bits. Thus the total complexity is less than 10,000 bits (compared to 268 *million* for RPS). The storage complexity for $kL = 2^{24}$ values, is 8 KB for k secrets and less than 128 MB for $k(L-1)$ public values. The number of operations with secrets is $\bar{m} = \xi^2 P = 128$ secrets, which is the same as RPS. A ready comparison of the complexities of (n,p)-secure RPS and I-HARPS for $n = 2^{16}$, $p < 10^{-20}$ is tabulated below (both need roughly the same number of block cipher operations, $m = 128$).

Complexity	RPS	I-HARPS
Public function	268 million	10,000
Total storage	64 MB	128 MB

Like all PKPSs, the security if I-HARPS degrades gracefully. The same $(2^{16}, 10^{-20})$-secure I-HARPS can resist coalition of upto 145,000 nodes with $p \approx 10^{-6}$ (or an attacker who has exposed all secrets from 145,000 devices can discover one in a million SAs).

D. Deployment Issues

With any key distribution scheme involving a KDC who escrow (or can calculate) all secrets, compromise of the KDC, or misuse of powers by the all powerful KDC, is an undesirable possibility. Fortunately (with any PKPS) it is trivially possible for the KDC to be split into independent entities. In fact, the KDC could consist of P different trusted authorities, each using a single master secret and generating $\binom{L}{2}$ public values, and making the public values available in public repositaries for download.

Note that nodes do not even need to authenticate themselves to download (and store) public values. Thus the public values could even be made available in *untrusted* public repositories that anyone can access (or even say distributed in optical storage devices that can be picked up near payment counters in super-markets). However, while modification of public values will not result in compromise of secrets, they can result in a lot of inconvenience (as nodes cannot establish security associations). In the LM-KDS scheme, Leighton and Micali [2] suggest a complete secondary KDS exclusively for

1-4244-0357-X/06/$20.00 ©2006 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.

authentication of public values of the primary KDS. However this may be an overkill.

Only nodes that have the secret $K_i^a = h(K_i, a)$ or $K_i^b = h(K_i, b)$ make use of the public value $P_i^{a,b}$. Thus for the convenience of permitting downloads of public values from untrusted public repositaries, for each public value, two message authentication codes W_i^a and W_i^b could be appended. Specifically, W_i^a is the authentication code for $P_i^{a,b}$ created using (a fixed function of) K_i^a, and W_i^b, likewise, employing a function of K_i^b.

Furthermore, the MACs W_i^a and W_i^b need not be the same size as the keys themselves. For instance the keys and public values could be 128 bits. While the MACs that are *computed* could be 128-bits, only a few (say b) LSBs may be appended for authentication. Note that without the knowledge of the secret used for authentication, any one who tries to alter the public value has only one viable option - guess the b-bit MAC - which will be detected by the verifier with a probability $1 - 1/2^b$ (for example 0.99609 for $b = 8$). Any end user who receives inconsistent public values from a repository will tend to avoid such repositories in the future. Thus there is no motivation for providers of such public repositories to take the risk of guessing MACs. Even $b = 8$ or $b = 16$ may be more than adequate.

For public values distributed in optical storage discs (in supermarkets) each disc could contain t folders, each with $\binom{L}{2}$ public values corresponding to t out of P LM-KDS systems. For each of the t folders, for example the folder corresponding to the master secret K_i, will include L message authentication codes (using secrets $K_i^1, K_i^2, \ldots, K_i^L$) for the entire set of $\binom{L}{2}$ values.

IV. Conclusions

We introduced a novel key distribution scheme, I-HARPS, which makes very good use of the most inexpensive of resources, storage, and consequently results in significant reduction in computational complexity, and gains in the achievable security of key pre-distribution schemes.

The reason for the improvement of I-HARPS over RPS is an advantageous trade-off. By choosing an independent parameter L, I-HARPS permits reduction in secure storage (or storage for secrets) by a factor $\gamma = 2/L$ and reduction in computational complexity by a factor γ. The price paid is the doubling of required storage. Consequently, the computational complexity of I-HARPS can be made as small as we desire, and the security of I-HARPS is only constrained by storage - a resource with perhaps the fastest Moore's law growth rate. With SD cards of upto 8 GB already in the market, and with no end in sight of saturating Moore's law (at least for storage), even storage of a few GBs in each node may be considered "free" (if not now, surely in the very near future). Thus I-HARPS that can resist even millions of colluding nodes, without any increase in computational complexity, is very much practical.

While KPSs have received substantial attention in the recent past mainly in the context of highly resource constrained sensor networks, I-HARPS *expands the scope* of possible uses of KPSs. After all, I-HARPS can easily withstand collusions of even hundreds of thousands of nodes with very little computational complexity and very reasonable storage requirements (a few hundred MBs). In any practical application scenario, even with very simple mechanisms for protection of secrets - for example if the secrets stored in devices (like PDAs/ mobile phones) are protected by passwords or biometric authentication (to prevent attackers from exposing secrets from stolen devices) - it is indeed inconceivable that an attacker can enlist hundreds of thousands of *willing* colluders (even for network sizes of the order of billions).

Furthermore, as the storage need not be trusted, even access over insecure networks to storage resources may be sufficient. In most application scenarios, including wireless sensors without physical access to large storage devices, even for scenarios where *both* interacting nodes do not have physical access to storage, it is conceivable that one of the sensors have access to some central facility over a (possibly insecure) network. The public values could be sent over such networks *on demand*. It is important to note that while a large number of public values may be *stored*, only $m \approx 128$ values are *needed* for evaluating any particular SA. Thus even the bandwidth overheads that are called for (in case direct physical access to storage is not possible) is low.

References

[1] M. Dyer, T. Fenner, A. Frieze and A. Thomason, "On Key Storage in Secure Networks," *Journal of Cryptology,* **8**, 189–200, 1995.

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

[3] Web Link, http://www.ietf.org/html.charters/manet-charter.html

[4] D. Boneh, M. Franklin, "Identity-based encryption from the Weil pairing," Advances in Cryptology – Crypto'2001, Lecture Notes on Computer Science 2139, Springer-Verlag (2001), pp. 213–229.

[5] 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.

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

[7] C.J. Mitchell, F.C. Piper, "Key Storage in Secure Networks," *Discrete Applied Mathematics,* **21** pp 215–228, 1995.

[8] P. Erdos, P. Frankl, Z. Furedi, "Families of Finite Sets in which no Set is Covered by the union of 2 Others," *Journal of Combinatorial Theory, Series A,* **33**, pp 158–166, 1982.

[9] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, B. Pinkas, "Multicast Security: A Taxonomy and Some Efficient Constructions," INFOCOMM'99, 1999.

[10] 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.

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

[12] M. Ramkumar, N. Memon, R. Simha, "Pre-Loaded Key Based Multicast and Broadcast Authentication in Mobile Ad-Hoc Networks," Globecom-2003, San Fransisco, CA, Dec 2003.

[13] M. Ramkumar, N. Memon, "An Efficient Random Key Pre-distribution Scheme for MANET Security," IEEE Journal on Selected Areas of Communication, March 2005.

[14] S. M. Matyas, C. H. Meyer, "Generation, Distribution and Installation of Cryptographic Keys," IBM Systems Journal, **2**, pp 126 – 137, 1978.

1-4244-0357-X/06/$20.00 ©2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.