On the Scalability of an Efficient “Nonscalable” Key Distribution Scheme

Mahalingam Ramkumar
Department of Computer Science and Engineering
Mississippi State University.

Abstract— **Some of the important requirements of key distribution schemes for many emerging applications involving resource constrained devices (for example, ad hoc and sensor networks) are 1) low complexity; 2) the ability to support unmediated establishment of security associations and 3) scalability. We propose a novel key distribution scheme based on an elegant scheme proposed by Leighton and Micali [1]. While the scheme in [1] was intended to be an alternative to Kerberos (employing a trusted server for mediation), the proposed modifications are towards realizing a scheme for non-mediated establishment of pairwise secrets. We enumerate several compelling advantages of the proposed scheme over existing schemes in the literature.**

I. Introduction

Rapidly lowering costs of computing and communication devices is expected to result in explosive growth of mobile computers equipped with wireless capabilities. Notwithstanding advances in semiconductor technology and significant improvements processing speeds, mobile devices are often constrained by battery life. It is very much desirable to reduce the computational and bandwidth overheads to be borne by such devices to prolong their battery life.

An important requirement for securing any network is the ability for participants to establish cryptographic security associations. Such cryptographic associations are facilitated through key distribution schemes (KDS). While KDSs based on asymmetric cryptographic techniques have the advantage of unconstrained scalability, this comes at the cost of non-negligible bandwidth and computational overheads, which is especially disadvantageous for mobile devices. Schemes based on purely symmetric cryptographic primitives, unfortunately, do not scale very well. More specifically, scalable schemes based on purely symmetric primitives either a) require a trusted server to be on-line for mediation of security associations, or b) are susceptible to collusions.

Schemes that simultaneously overcome the need for a mediator *and* susceptibility to collusions do not scale well. For example the “basic” key predistribution scheme requires each node requires to store $N-1$ secrets for a network size of N.

A. Contributions of this Paper

For many practical networks the total number of subscribers may be limited. For example, for networks within the scope of a large organization, the total number of subscribers may rarely exceed a few millions. While computational and bandwidth overheads are expensive for mobile devices, storage is an inexpensive resource. Almost any conceivable network enabled mobile device can afford several GBs of pluggable flash storage. Given that storage for a million 64-bit secrets is a mere 8 MB, for many networks the need for $\mathbb{O}(N)$ storage may not be the limiting factor.

Unfortunately, even in scenarios where $\mathbb{O}(N)$ storage requirement is not a limiting factor, scalable key distribution schemes are preferable, as term “scalability” refers to much more than simply the “number of subscribers” that a network can support. Scalable key distribution schemes typically permit seamless induction of new subscribers into the network, and include mechanisms for binding a unique subscriber identifier with the credentials of the user (for example, useful information like real-life identities, and special privileges / restrictions associated with the subscriber).

In this paper we propose a novel ID-based key distribution scheme which relies only on efficient symmetric cryptographic primitives. While the proposed scheme is limited in the total number of subscribers it can support (up to a few tens of millions in practice), it has many of the other advantageous features of scalable key distribution schemes. The scheme proposed in this paper is a modification of an elegant scheme proposed by Leighton and Micali [1] as an alternative to Kerberos. However, unlike the scheme in [1], the modified Leighton-Micali scheme (MLS) proposed in this paper does *not* require an on-line server for mediation. Apart from possessing some of the useful features of scalable schemes like *seamless induction of new subscribers*, the MLS scheme also has some other desirable features like the *ability to support multiple escrows* without increasing overheads.

In Section II of this paper we discuss the problem of key distribution and provide a broad overview of current approaches for this important problem. We compare and contrast three families of scalable key distribution schemes and discuss their limitations for networks that include resource constrained mobile devices. We then investigate the applicability of the non scalable “basic” key predistribution scheme, its limitations, and overheads necessary to overcome its limitations. The MLS scheme is proposed in Section III and its advantages are enumerated. Conclusions are offered in Section IV.

II. Key Distribution Schemes

A key distribution scheme (KDS) is a mechanism for distributing secrets and / or public values to all participants (or entities or subscribers) of a network, to facilitate establishment of cryptographic bonds or security associations (SA) between the entities. The most common of such SAs are a) one-to-one SAs in the form of pairwise secrets, for mutual authentication, and b) one-to-many SAs for source authentication. In this paper we restrict ourselves to key distribution schemes for establishment of shared secrets between every pair of entities.

A. KDS Requirements

A key distribution scheme for a network is controlled by one or more key distribution centers (KDC) who *induct* subscribers into the network by issuing secrets and / or certificates to the subscribers. Every subscriber is associated with a unique identifier (ID) and a description of his / her credentials. For example, a subscriber Alice may be associated with some ID A and credentials S_A. The credentials S_A may consist of brief descriptions of pertinent information like real-life identity of Alice, special privileges granted and / or restrictions imposed by the controllers of the network, and duration of validity of the credentials.

The fundamental expectation from the key distribution scheme is to permit any two subscribers of the network to 1) establish a shared secret, and 2) verify each other's credentials. Two entities A and B can use the shared secret K_{AB} to establish a private channel between A and B. Entity A can also authenticate a message M to B (and vice-versa) by appending a hashed message authentication code (HMAC) $h(M, K_{AB})$. Apart from catering for the two fundamental requirements, some of the other desirable features of a KDS include

1) Ability to support new subscribers to join the network at any time;
2) Reduce or avoid the need for frequent interactions between the KDC and subscribers. Ideally, the subscribers should need to interact with the KDC only during their initial induction into the network, and for renewal of secrets.
3) Low computational and bandwidth overheads for establishment of shared secrets. This is especially desirable for many emerging applications involving mobile devices.
4) While for many practical deployments the network may not require to support an unlimited number of subscribers, the ability to do so is a desirable feature.

In the rest of this section we review some of the existing approaches for the problem of key distribution, and enumerate their limitations for networks that may include resource constrained devices.

B. Scalable Schemes

Key distribution schemes can be broadly classified into scalable and non-scalable schemes. An example of a non scalable scheme is the "basic" key predistribution scheme (basic KPS), where for a network with N subscribers, the KDC chooses $\binom{N}{2}$ pairwise secrets and provides each subscriber with $N-1$ secrets. Scalable schemes include Kerberos-like schemes based on the symmetric Needham-Schroeder protocol [2]. Due to the need for a trusted server for mediation, Kerberos-like schemes may be ill-suited for many application scenarios. Scalable schemes that do not require a mediator, or schemes which facilitate *ad hoc* establishment of shared secrets, can be classified into certificates based schemes and identity based schemes.

1) Certificates Based Schemes: In certificates based schemes the KDC acts as certificate authority (CA). The CA generates a public and private key pair (U_T, R_T) of some asymmetric cryptographic scheme (for example, RSA). Every subscriber generates a key pair. For example, the subscriber Alice, associated with credentials S_A, generates a key pair (U_A, R_A). Inducting the subscriber Alice into the network is performed by the CA, by issuing a certificate C_A binding the public key U_A and the credentials S_A of Alice as $C_A = \langle U_A, S_A \rangle_T$. The certificate C_A can be verified by any subscriber with access to a genuine copy of the public key U_T of the CA. No further involvement of the CA is mandated for the day to day operation of the network. Any two subscribers, say Alice and Bob, can exchange their respective public key certificates C_A and C_B and perform some asymmetric computations computations to establish a shared secret K_{AB}.

2) Identity Based Schemes: For ID-based schemes the ID A of a entity is typically a one-way function of the credentials S_A. The ID $A = f(S_A)$ also doubles as the public key, thus obviating the very need for certificates. In ID-based schemes the KDC chooses public parameters of the system and one or more *master* secrets. The KDC can compute the private key(s) corresponding to *any* public key (ID). The private key R_A for a subscriber with identity $A = f(S_A)$ is thus *assigned* by the KDC to the subscriber A. Two subscribers A and B can independently compute a shared secret K_{AB} with implicit authentication of their IDs. If required, they can also exchange their credentials (preimages of the ID) and verify the credentials against the authenticated IDs.

ID-based public key schemes for encryption (IBE) and signatures (IBS), most of which take advantage of pairings in special elliptic curve groups, have attracted substantial attention recently [4].

3) Scalable ID-based KPS: In ID-based *key predistribution schemes* (KPS) [5], [7], the KDC chooses a set of P master secrets $\mathbb{S}$. Each subscriber is provided with a set of $k \leq P$ secrets. The set of k secrets $\mathbb{S}_A$ assigned to the subscriber Alice with ID $A = h(S_A)$ is determined by a public function which takes two inputs - the ID A, and the set of P KDC secrets $\mathbb{S}$. Two subscribers A and B (with secrets $\mathbb{S}_A$ and $\mathbb{S}_B$ respectively) can independently compute a pairwise secret K_{AB}.

However, an attacker who has access to secrets of many (say v) subscribers $\{O_1 \cdots O_v\}$ may be able to discover K_{AB} even *without* access to secrets $\mathbb{S}_A$ or $\mathbb{S}_B$. An n-*secure* KPS can resist collusions of up to n subscribers pooling their secrets together. Most n-secure scalable KPSs mandate $\mathbb{O}(n)$ storage and computational overheads, *irrespective* of the network size N [7].

4) Comparison of Scalable Schemes: ID-based schemes are increasingly seen as preferable over certificates based schemes for large scale networks, and especially for many emerging application scenarios like ad hoc networks. In existing networks based on the client-server paradigm the overheads for establishing a shared secret between a client and a server (bandwidth overheads for exchange of certificates and computational overheads for their verification) can be amortized over a large number of packets exchanged between them subsequently. In ad hoc networks where nodes typically exchange small packets with a large number of nodes, the overheads cannot be amortized.

With ID-based schemes two subscribers A and B can *independently* compute a shared secret K_{AB} with just the knowledge of their IDs. That they can establish a shared secret provides an assurance to A that the entity at the other end is B (and vice-versa).

In most scenarios A may not require "finer details" about B: like (for example) B is associated with credentials $S_B =$ "Bob Cryptographer, Anytown, USA, ...". Just the fact that A and B can establish a secret K_{AB} is enough to convince A that B (who ever B may be) "belongs to the network" (and vice-versa). For example, in a mobile ad hoc network the ID B may simply be the network ID of a mobile device. While a neighbor A of B will need to have some verifiable proof of the network ID of B, A does not need to know the consummate set of credentials associated with B.

In other words in ID-based schemes exchanging credentials is optional. However exchange of certificates is mandatory in certificates based schemes. Note that in certificates based schemes the public key U_B offers no indication that B "belongs to the network" unless A is able to verify the certificate C_B.

A disadvantage of identity based schemes is that they are inherently escrowed (as the KDC can compute secrets of all subscribers). Thus in scenarios where an all-powerful KDC is not acceptable, the ability to use multiple independent KDCs is a desirable feature.

For all three schemes discussed above subscribers can be inducted into the network at any time. The values provided to a subscriber A will permit the subscriber to establish a shared secret and verify the credentials of any subscriber B, even if B was inducted into the network after A was issued secrets. Their specific disadvantages are as follows:

1) Certificates based asymmetric schemes call for substantial computational and bandwidth overheads;
2) ID based public key schemes demand substantial computational overheads; and
3) ID based KPSs are susceptible to collusions, and thus may be acceptable only in scenarios where proactive measures are in place to protect compromise of secrets.

C. Non Scalable Key Predistribution

Unlike scalable KPSs, non scalable KPSs like the basic KPS are *not* susceptible to collusions. Furthermore, they require only $\mathbb{O}(1)$ computational overheads. However they demand $\mathbb{O}(N)$ storage overheads for a network size of N.

Fortunately many practical networks will *not* need to support an *unlimited* number of subscribers. Examples of such networks include those limited to members of an organization or group or participants in some a geographical region or entities with similar functions. Even a network which includes every member of a large organization like the US military (about 1.5 million on active duty and 3 million including reserves) does not need to cater for an unlimited number of participants.

Furthermore, storage is rapidly becoming an inexpensive resource, even for mobile computing applications. Given the fact that most mobile devices can afford several GBs of storage, and that many practical networks may not really need to support more than a few million or a few tens of millions of subscribers, the $\mathbb{O}(N)$ storage requirement may not be a limiting factor for many useful application scenarios.

While at first sight the basic KPS may seem adequate for most practical networks, the term "scalability" refers to more than just the number of subscribers that can be supported. Note that for all the scalable schemes discussed thus far (certificates based schemes, ID based public key schemes and ID-based KPSs) adding new subscribers to the network could be accomplished seamlessly. However this is not as easily feasible for the basic KPS.

Consider a scenario at some time t, when the network consists of $\bar{N}$ inducted subscribers. Assume that a new subscriber Y is added to the network. While the new subscriber Y can be provide with $\bar{N}$ secrets corresponding to the $\bar{N}$ existing subscribers, it is indeed cumbersome to provide a new secret corresponding to Y to each of the $\bar{N}$ existing subscribers.

1) Overcoming Limitations of Basic KPS: One strategy to overcome this limitation of basic KPS is by "over-provisioning," and the use of certificates. Let us assume that the network operators desire to cater for a maximum of $N_{max} = 2^{24}$ (about 16 million) subscribers. The KDC chooses a master secret K. An entity Alice described by credentials S_A is assigned a 24-bit KDS ID $1 \leq a \leq N_{max}$, and provided with $N - 1$ secrets

$$K_{aj} = h(K, a, j) \oplus h(K, j, a),$$

$\forall j \in \{1, 2, \ldots, N_{max}\} \setminus a$. In addition, the KDC provides A with a certificate binding a and S_A. Alternately, in some practical application scenarios, for example in a scenario where every entity corresponds to a hand held device that could take part in some network, the credentials S_A may simply be a unique network ID A. In such scenarios the certificate will bind the ID A with the KDS ID a.

If all secrets are 80 bits long, each subscriber will require storage of 160 MB for the secrets. Any two subscribers, irrespective of when they joined the network, can establish a shared secret and verify each other based on the 24-bit KDS ID and exchange certificates binding their KDS IDs to their credentials (or network IDs).

If we wish to avoid asymmetric primitives, certificates can be signed by the KDC using one-time signatures [8], [9]. As every one-time signature is associated with a unique public commitment, such commitments have to be distributed to all subscribers. Distribution of S commitments to the verifiers can be realized at the cost of $\log_2 S$ bandwidth overheads, using a Merkle hash tree [10]. For example, a Merkle tree of depth 24 can be used to commit 2^{24} (about 16 million) values. If every verifier stores only the root of the tree, 24 hashes will need to be supplied to verify any commitment. Some bandwidth vs storage trade-offs are also possible. As storage is relatively inexpensive each verifier can be provided with 2^{22} values of row 22 of the Merkle hash tree. The in-network bandwidth overheads for authenticating the commitments can be reduced from 24 hashes to $24 - 22 = 2$ hashes. The storage required is 40 MB (for 2^{22} 80-bit hashes).

In practice N_{max} may need to be substantially larger to than the actual maximum expected network size N to cater for unforeseen growth of the network (say $N_{max} = \nu N$ with a "safety factor" of $\nu > 1$). Thus modifying the basic KPS to facilitate seamless addition of subscribers into the network demands 1) storage for $N_{max} > N$ secrets 2) bandwidth overheads for exchanging certificates signed using one-time signatures 3) storage / bandwidth overheads (depending on the trade-offs employed) for distribution of N_{max} OTS commitments.

In Section III we propose an ID-based non-scalable KPS which

1) requires only $\mathbb{O}(1)$ computational overheads (like the basic KPS);
2) requires lower storage compared to the basic KPS for the same network size;
3) permits seamless addition of new nodes into the network; and
4) does *not* require certificates for binding credentials with identities.

III. Proposed Scheme

In [1] Leighton and Micali proposed an alternative to Kerberos in which the KDC chooses a master key K and a hash function $h()$. Subscriber A is provided with the secret $K_A = h(K, A)$. When A desires to establish a session secret with B, A approaches the *on-line* KDC to receive a public value P_{AB}

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

The shared secret between A and B is computed by A as

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

Note that B can also compute the shared secret $K_{AB} = h(K_B, A)$ directly using its secret K_B (*without* using the public value P_{AB}). Now A can choose a random session secret K_s and convey it to B by encrypting it using the secret K_{AB}. Similarly, when the initiator is B, the shared secret K_{BA} is computed by A as $h(K_A, B)$ and by B as $h(K_B, A) \oplus P_{BA}$.

Note that the value P_{AB} reveals no information about the secrets of A and B or the pairwise secrets K_{AB} or K_{BA}. Such values are referred to as "public" as they do not need to be kept a secret.

A. LM-KDS to LM-KPS

If every node is provided with public values corresponding to all other nodes, the LM-KDS becomes a key *predistribution* scheme (LM-KPS) and permits ad hoc establishment of pairwise secrets. Thus for a network consisting of N subscribers, each subscriber will need to store $N - 1$ public values (and one secret). Note that for the basic KPS for a network of N nodes, each node requires to store $N - 1$ *secrets*.

Though storing $N - 1$ public values (which need not be protected) seems more appealing than storing $N - 1$ secrets, in practice the advantage is not significant. After all, in any scenario where multiple secrets need to be stored, it is always common practice [11] to encrypt all secrets using a single well-protected secret and store the encrypted secrets in unprotected or less protected locations. Secrets stored in unprotected locations need to be authenticated and encrypted. Public values stored need to be authenticated (for example by appending a message authentication code based on a protected secret). However, in the rest of this section we propose some simple modifications to the Leighton-Micali scheme to significantly improve its utility.

1) LM-KPS to MLS: The crux behind the modified LM scheme (MLS) comes from the realization that to establish a shared secret between A and B, only one of them require access to the public value $P_{AB} = P_{BA}$. As long as a clear rule exists for A and B to unambiguously determine *who* needs to employ the public value, both of them can compute the shared secret (either as $h(K_A, B)$ or as $h(K_B, A)$). This trivial modification has three non-trivial, and highly desirable side effects:

The obvious side-effect, is the fact that storage can be halved as each subscriber will need to store only $N/2$ public values (for a network of size N).

Secondly, recall that the reason that the basic KPS does not facilitate seamless addition of new subscribers is that it is impractical to distribute keys corresponding to new subscribers to old subscribers. To overcome this issue we had to provide old subscribers with additional secrets in anticipation, reserved for subscribers who may be inducted in the future. With the modified LM-KPS the new subscribers can be provided with public values to establish shared secrets with old (existing) subscribers, as only one of the two need access to the public value. The old subscribers need not be provided with any values in anticipation of subscribers who may join in the future. Thus the need for over-provisioning is eliminated, which *further* reduces the storage requirement.

Even while (in the modified basic KPS) over provisioning facilitated establishment of shared secrets between new and old subscribers, at the time when the keys (corresponding to anticipated subscribers) are actually provided to existing subscribers, the credentials of the future subscribers are unknown. It is for this reason that subscribers had to be associated with some unique KDS ID that is no way related to their credentials, and certificates had to be issued by the KDC to bind credentials with KDS IDs. With the modified LM scheme we do not need certificates any more. The secret issued to an entity can be directly tied to the credentials (or an ID which could be a one-way function of the credentials). Thus the modified LM scheme is an ID-based scheme.

B. Modified Leighton Micali Scheme

The assumed network model is as follows. The network consists of one or more KDCs who induct subscribers into the network by providing them with secrets, over a secure channel. For simplicity we shall first assume a single KDC. Later in this section we shall discuss extensions to support multiple independent KDCs.

The subscribers can receive public values over any open network. For example, such parameters could be conveyed through optical discs over postal networks or downloaded from public FTP sites.

Assume that the N_0 subscribers were inducted into the network before time T_0. At time T_0 the KDC "prepares" the public values for delivery to the N_0 subscribers. As an example, the process of providing public values to a subscriber A may consist of the following steps:

1) preparing a file A.dat containing all public values necessary for the subscriber A;
2) uploading the file to an FTP site;
3) conveying the URL of A.dat to the subscriber A (say, via Email).

At regular intervals of time thereafter (for example, annually) the KDC repeats the process of delivering public values to all its existing subscribers. We shall denote such regular instances of time as $T_i = T_0 + i\Delta$, and refer to the time between any two such instances T_i and T_{i+1} as an "epoch."

Let N_i represent the number of inducted subscribers before time T_i. During the $(i+1)^{\text{th}}$epoch (between T_i and T_{i+1}) we refer to the N_i subscribers as "continuing subscribers" and the subscribers inducted after time T_i (and before time T_{i+1}) as "new subscribers."

1) Subscriber IDs: The KDC chooses a master secret K. A subscriber described by credentials S_A is associated with the ID $A = h(S_A)$, and receives the secret $K_A = h(K, A)$ from the KDC over a secure channel.

If identities as chosen as one-way functions, reasonably large IDs are required to ensure that collisions are rare. For networks expected to support up to say 2^t subscribers, the IDs should be substantially larger than $2t$-bits. Furthermore *pre-image* resistance of the hash function $h()$ is also required to ensure that A cannot substitute the pre-image S_A with an alternate one - say S'_A where $A = h(S'_A)$. As the proposed scheme is meant for network sizes of at most few tens of millions, 80 bit hashes are more than adequate.

Even while a rich set of affiliations / restrictions can be indicated in the credentials of participants and bound to their IDs, such credentials can be conveyed only after some bandwidth overheads for releasing the pre-image (credentials). As it is indeed preferable to reduce the need for the overheads for this purpose, some commonly used credentials could be directly indicated using special bits of the ID. Let us assume that the ID of every entity is 128 bits long, where 80 bits are derived as the hash of the credentials and 48 bits are used for "other" purposes.

In the MLS scheme 24 of the 48 bits are used for dealing with new subscribers (such bits are all zeros for continuing subscribers), leaving 24 bits for other purposes. For example, some bits can be used to indicate validity duration (if the subscriber's credentials expire before the end of the current epoch), if the entity is a minor, if the entity is a guest member, membership gradations like silver / gold / platinum, etc.

2) Continuing Subscribers: Let us represent by $E_1 \cdots E_{N_i}$ the identities of the N_i continuing subscribers during the epoch between T_i and T_{i+1}. The public values provided to the entity E_j (with secret K_{E_j}) are of the form $[E_k, P_{E_j E_k}]$, where

$$P_{E_j E_k} = h(K_{E_j}, E_k) \oplus h(K_{E_k}, E_j). \tag{3}$$

More specifically, out of the N_i values corresponding to identities $E_1 \cdots E_{N_i}$ only $N_i/2$ values (on an average) are provided to the entity with ID E_j. The public value corresponding to the ID E_k is provided to subscriber with ID E_j if

1) $E_k < E_j$ and $E_k + E_j$ is even; or
2) $E_k > E_j$ and $E_k + E_j$ is odd.

In other words, a subscriber with even ID A stores public values corresponding to all even IDs less than A and all odd IDs greater than A. Similarly, a subscriber with an odd ID B stores public values corresponding to all odd IDs less than B and all even IDs greater than B.

3) New Subscribers: New subscribers, who join during the epoch between T_i and T_{i+1} are associated with an additional serial number. The j^{th}new subscriber is associated with a serial number j. This serial number is indicated as part of the ID of the entity (24 reserved bits to permit up to 16 million new subscribers to join during an epoch). For all existing subscribers $j = 0$. New subscribers receive public values corresponding to *all* entities inducted earlier. In other words, the j^{th}new subscriber receives $N_i + (j-1)$ public values corresponding to all N_i continuing subscribers and the $j-1$ new subscribers inducted before the j^{th}new subscriber.

4) Computing Shared Secrets: The rule for computing the shared secret K_{AB} between two entities with IDs A and B is as follows. If the 24 bits (indicating the serial number for new subscribers and 0 for all continuing subscribers) are different for the two subscribers, the subscriber with the higher serial number will use the public value. For example, if A is expected to employ the public value P_{AB} the shared secret is computed as

$$K_{AB} = h(K_B, A) = h(K_A, B) \oplus P_{AB} \tag{4}$$

If the 24 bits indicating the serial number are the same for A and B the applicable rule is as follows:

1) If $A+B$ is even then the node with the larger ID employs the public value P_{AB};
2) If $A+B$ is odd, the node with the lower ID employs the public value P_{AB}.

Note that, apart from eliminating the need for one-time signatures (as was required for extending the basic KPS to support seamless joins), the worst case storage (for the most recently inducted node) is the same as the number of the current subscribers. Furthermore during the next epoch (when new subscribers become continuing subscribers) they will need to store only $N_{i+1}/2$ values. More importantly, no hard limit need to be placed on the total number of entities who can join the network. The network size can keep increasing as much as storage capabilities permit.

5) Multiple KDCs: The MLS scheme can be easily extended to cater for multiple escrows without increasing the storage requirements. For example, if 2 KDCs are used each KDC chooses a master secret - say K^1, K^2 respectively, and every subscriber receives one secret from each KDC. Entity A receives secrets $K_A^1 = h(K^1, A)$ and $K_A^2 = h(K^2, A)$ respectively. A public value P_{AB} now takes the form

$$\begin{aligned} P_{AB} &= P_{AB}^1 \oplus P_{AB}^2, \text{where} \\ P_{AB}^i &= h(K_A^i, B) \oplus h(K_B^i, A), i = 1, 2. \end{aligned} \quad (5)$$

The 2 KDCs can compute and submit the public values of the form $P_{AE_j}^i$ meant for A to a central repository where they are XOR-ed together and delivered to A. In a scenario where A is required to use the public value P_{AB}, the shared secret K_{AB} is

$$\begin{aligned} K_{AB} &= h(K_B^1, A) \oplus h(K_B^2, A) \\ &= (h(K_A^1, B) \oplus P_{AB}^1) \oplus (h(K_A^2, B) \oplus P_{AB}^2) \\ &= h(K_A^1, B) \oplus h(K_A^2, B) \oplus P_{AB} \end{aligned} \quad (6)$$

Thus if t independent KDCs are used only additional overhead is the need for t hash function evaluations instead of one (for computing a pairwise secret).

IV. Conclusions

A few decades ago asymmetric primitives were prohibitively expensive for many computers. While the improved ability of computers have made the use of asymmetric primitives common-place, advances in technology have also simultaneously lowered the bar for network enabled devices. For many of the computers taking part in networks of the future, the use of asymmetric primitives may once again be impractical. However, unlike the scenario a few decades ago, storage is currently an inexpensive resource. This resource can be used advantageously to offset the limitations of symmetric schemes.

Certificates based and ID-based schemes relying on asymmetric primitives are unsuitable for many emerging application scenarios. While scalable ID based key predistribution schemes require low computational and bandwidth overheads, they have the disadvantage of susceptibility to collusions.

Even while the network scales that can be realized using non scalable schemes is storage limited, for many practical networks this may not be an issue, especially given the significant storage capabilities of mobile devices. The non scalable basic key predistribution scheme demands very low computational overheads, and is not susceptible to collusions. Unfortunately, it does not facilitate seamless addition of new subscribers. We argued that this limitation of basic KPS can be overcome only at the cost of increased storage overheads (by over-provisioning) and increased computational and bandwidth overheads (for certificates).

We then proposed a novel ID-based key distribution scheme which is comparable in complexity to the basic KPS, without being plagued by the many disadvantages of the basic KPS. While the total number of subscribers that can be supported by MLS is limited by storage capabilities, it has many of the advantageous features of truly scalable key distribution schemes like

1) the ability to seamlessly induct subscribers into the network;
2) the ability to permit verification of a rich set of credentials of subscribers without mandating overheads for certificates, and additionally,
3) permit multiple independent escrows without increasing the overheads.

References

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

[2] R. Needham and M. Schroeder, "Using encryption for authentication in large networks of computers," Communications of the ACM, 21(12), December 1978.

[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] T. Matsumoto, H. Imai, "On the Key Predistribution System: A Practical Solution to the Key Distribution Problem," pp 185–193. CRYPTO 1987.

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

[8] R. C. Merkle, "A Digital Signature Based on a Conventional Encryption Function," In Advances in Cryptology, Crypto 87.

[9] D. Bleichenbacher, U. Maurer, "On the Efficiency of One-Time Digital Signatures," ASIACRYPT: Advances in Cryptology : International Conference on the Theory and Application of Cryptology, 1996.

[10] R.C. Merkle "Protocols for Public Key Cryptosystems," In Proceedings of the 1980 IEEE Symposium on Security and Privacy, 1980.

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