Broadcast Encryption with Random Key Pre-distribution Schemes

Mahalingam Ramkumar
Department of Computer Science and Engineering
Mississippi State University, Mississippi State, MS 39762

Abstract

Broadcast encryption (BE) deals with the problem of establishing a secret, shared by $g = G - r$ *privileged* nodes, among a set G nodes. Specifically, a set of r *revoked* nodes are denied access to the secret. Many schemes to address this problem, based on key pre-distribution schemes (KPS), have been proposed in the literature. Most state-of-the-art methods employ tree-based techniques. However, *random* key pre-distribution schemes (RKPS), which have received a lot of attention in the recent past (especially in the context of ad hoc and sensor network security), also cater for BE. In this paper we analyze the performance of BE using RKPSs. While in most tree-based methods the source of the broadcast is assumed to be the root of the tree (unless asymmetric cryptographic primitives can be used), BE using RKPSs caters for BE by *peers* - without the need for asymmetric cryptography. Furthermore, unlike most BE schemes where the identities of the revoked nodes have to be explicitly specified, BE using RKPSs allow for protecting the identities of the revoked nodes, which could be a useful property in application scenarios where privacy is a crucial issue.

1 Introduction

Broadcast encryption (BE) [1] provides a means of establishing shared secret between g privileged nodes, among of a set of G nodes, where $g+r = G$, and the r nodes which are *not* provided with the secret are usually referred to a revoked nodes. For situations where $g << G$ it may be more efficient to set-up a shared secret between g nodes using g unicast transmissions. However, for scenarios where $r << G$ (or $g \approx G$) such an approach is very inefficient. BE schemes provide a very satisfactory solution for cases where $r << G$.

In many application scenarios [2], BE may assume a slightly different form. The universe, or the set $\mathbb{U}$, consists of all nodes in the system. Out of $\mid \mathbb{U} \mid = N$ nodes, there may exist a subset $\mathbb{G}_0 \in \mathbb{U}$ of G nodes. Typically, the G nodes share a secret (a priori), privy only to the nodes in the set $\mathbb{G}_0$. The problem that BE needs to address in this case, is the efficient dissemination of a secret to all nodes in $\mathbb{G}_0$ - except a subset $\mathbb{G}_R \in \mathbb{G}_0$ of r nodes. At the end, this results in a new subset $\mathbb{G}_1 = \mathbb{G}_0 \setminus \mathbb{G}_R$ of $g = G - r$ nodes, which now share a secret (not available to nodes not in $\mathbb{G}_1$).

Typically, BE is realized using some form of key pre-distribution, where a set of k secrets are distributed to each node in the universe of N nodes (before the system is deployed). The source of the broadcast then

1. chooses a broadcast secret K_b (intended for the set $\mathbb{G}_0$),
2. encrypts K_b using n keys $K_{e1} \cdots K_{en}$, and
3. transmits n values $E_{K_{ei}}(K_b), 1 \leq i \leq n$.

The keys $K_{e1} \cdots K_{en}$ are chosen in such a way that none of the r nodes in $\mathbb{G}_R$ can (using their preloaded secrets) "discover" *any* of the keys $K_{e1} \cdots K_{en}$, while the remaining $G - r$ nodes in $\mathbb{G}_1 = \mathbb{G}_0 \setminus \mathbb{G}_R$ may be able to discover *at least one* of the secrets $K_{e1} \cdots K_{en}$, and thereby gain access to the secret K_b. Typically, the source of the broadcast does *not care* if the nodes in $\mathbb{U} \setminus \mathbb{G}_0$ gain access to K_b. For example, if the G nodes shared a secret K_{N_0} *before* the broadcast, the shared secret between the nodes in $\mathbb{G}_1$ *after* the broadcast may be $K_b \oplus K_{N_0}$ - which neither the explicitly revoked nodes in $\mathbb{G}_R$ or the other nodes $\mathbb{U} \setminus \mathbb{G}_0$ can gain access to - the former do not have access to K_b and the latter do not have access to K_{N_0}.

The efficiency of BE schemes is usually measured in terms of:

1. The bandwidth needed for the broadcast. More specifically, the number of encryptions needed to securely convey the broadcast secret, and overheads, if any.
2. Resilience of the scheme to collusion of revoked nodes.
3. Storage complexity at the receivers of the broadcast.
4. Computational complexity for recovering the broadcast secret for each receiver.
5. Computational complexity involved in choosing the keys $K_{e1} \cdots K_{en}$ by the source of the broadcast.

Efficient solutions to the problem of broadcast encryption has received a lot of attention since the problem was defined by Fiat and Noar in Ref. [1]. Most current state of the art solutions [3] - [8] are tree-based, where the source of the broadcast is assumed to be the trusted authority (TA) at the root of the tree, who distributes the secrets in the first place. However, such schemes can generally be extended to permit broadcast by parties other than the TA - if asymmetric cryptographic primitives are employed.

In this paper we consider BE using random key pre-distribution schemes (RKPS). Though RKPSs can also be deployed in a tree-like hierarchy, we restrict ourselves to a "flat" deployment. We argue that BE using RKPS schemes offers many advantages over the better known tree-based BE schemes for many application scenarios. Specifically, the two primary advantages offered by RKPSs (over tree-based schemes) are:

1. they permit *any* node to perform BE, *without* the use of asymmetric cryptographic primitives
2. they permit revocation of nodes without *explicitly specifying* the identities of revoked nodes - which is potentially very useful when privacy is a concern.

المنارة للاستشارات
www.manaraa.com

The rest of this paper is organized as follows. In Section 2 we briefly review KPSs, with an emphasis on RKPSs. In Section 3 we provide a quantitative analysis of the efficiency of BE using RKPSs. Discussions, interpretations and comparisons with other BE schemes, and some potential applications is the topic of Section 4. Conclusions are offered in Section 5.

2 Key Pre-distribution

A KPS consists of a trusted authority (TA), and N nodes with unique IDs. The TA chooses P secrets $\mathcal{R}$ and two operators $f()$ and $g()$. The operator $f()$, is used to determine the secrets $\mathbb{A}$ that are preloaded in node A. Two nodes with IDs A and B, with preloaded secrets $\mathbb{A}$ and $\mathbb{B}$ can discover a unique shared secret K_{AB} using a *public* operator $g()$ without further involvement of the TA. The restrictions on the operators $f()$ and $g()$ in order to satisfy these requirements can be mathematically stated as follows:

$$\begin{aligned} \mathcal{S}_i &= f(\mathcal{R}, ID_i); \\ K_{ij} &= g(\mathcal{S}_i, ID_j) = g(\mathcal{S}_j, ID_i) \\ &= f(\mathcal{R}, ID_j, ID_i) = f(\mathcal{R}, ID_i, ID_j). \end{aligned} \tag{1}$$

As $g()$ is public, it possible for two nodes, just by exchanging their IDs, to execute $g()$ and discover a unique shared secret. As the shared secret is a function of their IDs, their ability to arrive at the shared secret provides mutual assurances to A and B that the other node possesses the necessary secrets $\mathbb{B}$ and $\mathbb{A}$, respectively. The secrets preloaded in each node is referred to as the node's *key-ring*. We shall represent by k, the size of the key ring.

The primary advantage of KPSs is their ability to cater for ad hoc authentication without active involvement of a trusted authority, and without employing asymmetric cryptography. However, this advantage comes at a price. Note that in KPSs, the keys assigned to different nodes are *not independent* - they are all derived from the same set (TA's) keys $\mathcal{R}$. Thus an attacker who has exposed keys from a *finite* number of nodes could compromise the entire system. For conventional key distribution schemes (KDS) (like Kerberos [10], [11] or PKI [12]) however, as the keys assigned to different nodes are *independent*, this is not the case.

However, for evolving [13] application scenarios (like MANETS [14]) where extensive *mutual co-operation* of resource constrained (battery operated) nodes is necessary for their very functioning, compromise of a few nodes could affect the entire deployment. Thus there is a need to take proactive steps to control sizes of *attacker coalitions* (perhaps by improved technology for tamper resistance / read-proofing of devices [15]). For securing such deployments, conventional KDSes may be an "overkill" (if the entire deployment is affected if a finite number of nodes are compromised, the fact that the *KDS* is not compromised does not help much). Thus KPSs, due to their inherent advantages of low resource consumption (which is also necessary as deployments of wireless devices forming MANETS are expected to include resource constrained devices), may be sufficient for securing such networks. This is perhaps the reason for renewed interest in KPSs in the recent past.

2.1 Random Key Pre-distribution Schemes

KPSs based on the concept of pre-loading subsets (say of cardinality k) of keys in each node, from a *pool* of P keys, has been employed by various researchers, for very different cryptographic primitives. Perhaps the earliest example is the matrix [16] key pre-distribution scheme by Gong et. al. The applications employing preloaded subsets range from discovery of shared secrets for pairwise communications [16] - [23], and more general group communications [22], [24], and broadcast authentication [26], [27], [28].

While the earlier methods based on preloaded subsets favored deterministic allocation of keys to nodes [29] (most of them perhaps motivated by Erdos et. al's seminal work on intersections of finite sets [25]), Dyer et al [27] was perhaps the first to point out the advantages of *random* allocation of subsets. A very elegant framework for analysis of the security of random preloaded subsets was also presented in [27]. Recent attempts in this direction too, [17] - [22] favor random [17], [18], [23] or pseudo-random [19], [22], allocation of keys (in this paper we shall collectively refer to them as RPS or random preloaded subsets). While all RPS based methods are essentially similar, the primary advantage of the methods which employ *pseudo-random* allocation of subsets, is that they provide a simple and elegant way for nodes to determine shared secrets (methods based on purely random allocation on the other hand need a bandwidth intensive shared key discovery process).

Formally, a (P, k) RPS employs a TA who chooses an indexed set of P keys $K_1 \cdots K_P$. Each node has a unique ID. The TA chooses public random function $F_{RPS}()$, which when "seeded" by a node ID, yields the allocation of keys for the node. Thus for a node A (node with unique ID A)

$$\begin{aligned} F_{RPS}(A) &= \{A_1, A_2, \ldots, A_k\}, \\ \mathbb{A} &= \{K_{A_1}, \ldots, K_{A_k}\}. \end{aligned} \tag{2}$$

where $1 \leq A_i \leq P, A_i \neq A_j$ for $i \neq j$. In other words $F_{RPS}()$ generates a *partial* random permutation of $\{1 \cdots P\}$. The k-length sequence $\{A_1, A_2, \ldots, A_k\}$ is the index of the keys preloaded in node A (or node with ID A). $\mathbb{A}$ is the set of secrets preloaded in A. Note that the indexes are public (as the node ID and $F_{RPS}()$ are public).

While KPSs employing random preloaded subsets fall under the category of *random* KPSs, the first RKPS, LM [30], proposed by Leighton and Micali, employs a very different idea. In the (k, L) LM scheme, the TA chooses an indexed set of k secrets $K_1 \cdots K_k$, a cryptographic hash function $h()$, and a public random function $F_{LM}()$. For a node A,

$$\begin{aligned} F_{LM}(A) &= \{a_1, a_2, \ldots, a_k\}, 1 \leq a_i \leq L \forall i. \\ \mathbb{A} &= \{{}^{a_1}K_1, {}^{a_2}K_2, \ldots, {}^{a_k}K_k\}. \end{aligned} \tag{3}$$

In other words $F_{LM}()$ generates a k-sequence of uniformly distributed random integer values between 1 and L. The node A

المنارة للاستشارات
www.manaraa.com

is preloaded with k keys. The i^{th} preloaded key is node A is derived by repeatedly hashing i^{th} TAs key a_i times. The parameter L is the maximum hash depth. The notation ${}^{i}K_j$ represents the result of *repeatedly* hashing of K_j, i times, using a (public) cryptographic hash function $h()$.

In HARPS [24], Ramkumar and Memon proposed a RKPS which is a generalization of LM and RPS. In (P, k, L) HARPS, the TA chooses P keys $K_1 \cdots K_P$, and each node is loaded with a *hashed* subset of k keys. The TA has an indexed set of P secrets, a cryptographic hash function $h()$ and a public random function $F_{HARPS}()$. For a node A,

$$\begin{aligned} F_{HARPS}(A) &= \{(A_1, a_1), (A_2, a_2), \ldots, (A_k, a_k)\}, \\ \mathbb{A} &= \{{}^{a_1}K_{A_1}, {}^{a_2}K_{A_2}, \ldots, {}^{a_k}K_{A_k}\}. \end{aligned} \tag{4}$$

The first coordinate $\{A_1, A_2, \ldots, A_k\}$ represents the index of the keys chosen to be preloaded in node A, and the second coordinate $\{a_1, a_2, \ldots, a_k\}$, the number of times each chosen key is hashed (using cryptographic hash function $h()$) before they are preloaded in the node A.

Note that LM and RPS are actually special cases of HARPS. LM is HARPS with $P = k$, and RPS is HARPS with $L = 0$ (or keys are not hashed before pre-loading).

3 Broadcast Encryption Using Random KPSs

As HARPS is a generalization of LM and RPS we shall only consider broadcast encryption using HARPS (the special cases easily follow). For BE using HARPS, the sender employs a subset of all secrets not covered by the union of the r revoked nodes. If we represent by $\mathbb{R}$ the entire set of secrets that the source has access to, and by $\mathbb{S}_r$ the secrets covered by the union of r nodes, each of the *independent* secrets in $\mathbb{R} \setminus \mathbb{S}_r$ can be used to encrypt the broadcast secret K_B.

Indexes	1	2	3	4	5	6	7	8
A	4	2	x	1	x	3	x	x
B	x	3	1	x	3	2	x	x
d_i	3	1	0	0	2	1	4	4
C	x	3	4	1	x	x	2	x
D	2	x	x	x	3	1	4	x

Consider the illustrative example above. The TA chooses $P = 8$ keys $K_1 \cdots K_8$, and $k = 4$, $L = 4$. In other words, each node is provided with a subset of $k = 4$ keys and each key has a "hash depth" between 1 and 4 (random and uniformly distributed). In the example, node A has keys with indexes $i = 1, 2, 4, 6$ at hash depths 4, 2, 1 and 3 respectively (or keys ${}^{4}K_1$, ${}^{2}K_2$, ${}^{1}K_4$ and ${}^{3}K_6$). The row d_i is the hash depths the TA *can* employ for each $1 \le i \le P$ for encrypting K_b which revokes A and B. TA can use keys ${}^{3}K_1$, ${}^{1}K_2$, ${}^{2}K_5$, ${}^{1}K_6$, ${}^{4}K_7$, and ${}^{4}K_7$.

Node C can use ${}^{4}K_7$ for decrypting the secret. Node D can use ${}^{3}K_1$ or ${}^{1}K_6$ or ${}^{4}K_8$. While the TA can use ${}^{0}K_3$ and ${}^{0}K_4$, it does not serve any purpose - no node can decrypt the broadcast secret encrypted with those keys.

On the other hand, if C is the *source* of the broadcast (revoking A and B) C could choose ${}^{4}K_8$ for encrypting K_b.

3.1 Analysis of Efficiency of Broadcast Encryption

Let us first consider the case of BE by the TA. The TA has access to all secrets $K_1 \cdots K_P$ (at hash depth 0). Each of the r (to-be-revoked) nodes have $k < P$ keys each. The hash depths of the keys are uniformly distributed between 1 and L. The union of indexes of keys in all r nodes may still *not* contain some of the P indexes. Obviously such keys can be used by the TA for encrypting the broadcast secret (as none of the r nodes can decrypt them). For example if the key index i is not present in the union of r nodes, the broadcast secret can be encrypted safely with ${}^{L}K_i$.

Now consider a key indexed i, which however, u of the r nodes have. Let us assume that the hash depths of those u keys are $d_1 \cdots d_u$, with $d_{min} = \min(d_1 \cdots d_u)$. The TA could still use key ${}^{d_{min}-1}K_i$ to encrypt the broadcast secret. Proceeding in this fashion, the TA could now use *on an average*, some n_j keys at each hash depth $1 \le j \le L$, for encrypting the broadcast secret.

With these $n = \sum_{j=1}^{L} n_j$ keys the TA hopes to "reach" *every* privileged node. Some of the transmitted keys may be useful to many nodes. Most may not be useful for a *particular* node. Once again, while the TA can encrypt the broadcast secret with keys $K_1 \cdots K_P$ (at hash depth 0), they are not useful for the purpose of reaching the nodes - none of the nodes can decipher them. Thus key indexes, where the corresponding $d_{min} = 1$ in the union of the r nodes, cannot be used by the TA for encrypting the broadcast secret.

However, while it is guaranteed that none of the r nodes (even if they pool all their secrets together) can decipher the broadcast secret, there is a possibility that some of the $g = G - r$ privileged nodes too may *not* be able to decrypt *any* of the n encryptions. We shall represent by p_o, the probability that an arbitrary node among the group of g previliged nodes *cannot* use *any* of the n keys. In such an event, on an average, gp_o privileged nodes may not be able to decrypt the broadcast secret. As long as $p_o << \frac{1}{g}$ this may not be a serious issue.

Note that it is also possible to trade-off p_o for bandwidth (n, the number of encryptions needed). For instance, the TA may decide to send only a subset of the encrypted secrets in order to meet a *target* p_o. For the gp_o nodes (on an average) that may be missed, the TA could send the broadcast secret using the secret the TA shares with every node (the secret shared between the TA and any node can be a function of *all* k keys in the node - as the TA has access to all secrets).

The question now is, how many secrets (on an average) does the TA need to transmit? But prior to that, we need an estimate the number of keys $n_j, 1 \le j \le L$ that the TA can use. It is easy to see that for $j = L$, the n_j keys correspond to the keys that none of the r nodes have (at *any* hash depth). The probability that any node has a key indexed i is $\xi = \frac{k}{P}$. Thus the probability that none of the r nodes have key i is $(1 - \xi)^r$. In other words

$$n_L = P(1 - \xi)^r. \tag{5}$$

المنارة للاستشارات
www.manaraa.com

Now let us evaluate the expression for n_j for a general j. Let us assume that u out of r nodes have some key index i (the probability that exactly $u > 0$ out of r nodes have the i^{th}key is $\binom{r}{u}\xi^u(1-\xi)^{(r-u)}, 1 \le u \le r$), with corresponding hash depths $d_1 \cdots d_u$. Under this condition, the TA can employ hash depth j for the key i if $d_{min} = \min(d_1 \cdots d_u) = j+1$. As each $d_l, 1 \le l \le u$ is uniformly distributed between 1 and L

$$\begin{aligned}\Pr\{d_{min} = j+1\} &= \Pr\{d_{min} > j\} - \Pr\{d_{min} > j+1\} \\ &= \frac{(L-j)^u - (L-j-1)^u}{L^u} \qquad (6)\end{aligned}$$

Thus the probability π_j that the TA employs hash depth j for key i is

$$\pi_j = \sum_{u=1}^{r}\binom{r}{u}\xi^u(1-\xi)^{(r-u)}\frac{(L-j)^u - (L-j-1)^u}{L^u}, \tag{7}$$

and

$$n_j = P\pi_j. \tag{8}$$

In order to decrypt a secret encrypted with key index i at depth j, the node should have the secret i (probability ξ) at depth $d \le j$ (probability $\frac{j}{L}$). Encryption keys that use higher hash depths are thus “more useful.” Thus for a particular encryption key at hash depth j (or any one of the n_j keys) the probability of outage is $p_{o_i} = (1 - \xi\frac{j}{L})$.

The TA does not have to use *all* $n = \sum_{j=1}^{L} n_j$ keys. The TA may instead only use a subset $m = \sum_{j=q}^{L} n_j$ keys. In the case the probability of outage for any node (or the probability that it cannot decipher *any* of the $\sum_{j=q}^{L} n_j$ encryptions) is

$$p_o^* = \prod_{j=q}^{L}(1 - \xi\frac{j}{L})^{n_j} \tag{9}$$

The total number of encryption needed to convey the secret to all $g = G - r$ nodes is therefore

$$n_e = \left(\sum_{j=q}^{L} n_j\right) + gp_o^*. \tag{10}$$

The source (TA in this case) would first try to use all keys at depth $j = L$. If that does not yield a satisfactory p_o the TA would then try adding the keys at depth $j = L - 1$, and so on. Thus if the broadcast uses a minimum depth of q it implies all possible encryption keys *above* depth q *will be chosen*. However, it is *not* generally necessary that *all* possible Keys *at* depth q are chosen. Out of the n_q possible keys only n_q^* keys may be chosen.

If the probability of outage is p'_o after all possible keys $\sum_{j=q+1}^{L} n_j$, above depth q are chosen (or $p'_o = \prod_{j=q+1}^{L}(1 - \xi\frac{j}{L})^{n_j}$), then the choice of $n_q^* \le n_q$ would result in an overall bandwidth of

$$n_e = \left(\sum_{j=q+1}^{L} n_j\right) + n_q^* + gp'_o\left(p_{o_q}\right)^{n_q^*}, \tag{11}$$

where $p_{o_q} = (1 - \xi\frac{q}{L})$. The optimal choice of n_q^* is therefore

$$n_q^* = \frac{\log\left(\frac{-1}{gp'_o\log(p_{o_q})}\right)}{\log\left(p_{o_q}\right)}, \tag{12}$$

resulting in an overall minimum bandwidth (number of encryptions of the broadcast secret needed for the broadcast) requirement of

$$n_e^* = \left(\sum_{j=q+1}^{L} n_j\right) + n_q^* + gp'_o\left(p_{o_q}\right)^{n_q^*}. \tag{13}$$

As HARPS is a generalization of RPS [22] and LM [30], extension of the results above to LM and RPS are trivial. For LM, $\xi = 1$. Which implies $n_L = 0$ and

$$n_j = \frac{(L-j)^r - (L-j-1)^r}{L^r} \text{ (for LM scheme)} \tag{14}$$

For RPS, all KPS keys (the P TA’s secrets and the k secrets in each node) have the same hash depth. so only keys that none of the r nodes have (which occurs with probability $(1-\xi)^r$ can be used - or $n = P(1-\xi)^r$. The probability of outage in this case is $p_o = (1-\xi)^n$. Once again, not all n keys may be needed. Only $q < n$ keys may be chosen in order to minimize

$$n_e^* = q + g(1-\xi)^q, \tag{15}$$

Or

$$q = \min\left(\frac{\log\left(\frac{-1}{g\log(1-\xi)}\right)}{\log(1-\xi)}, n = P(1-\xi)^r\right). \tag{16}$$

Extension of the analysis above to broadcast encryption by *peers* is also trivial. Note that if the source is a peer, it may not be able to use all possible n_j keys at depth j that the TA can. It can use a key at depth j only if the source node *has* the key (probability ξ) *and* even if has a key for that index, the hash depth of the key should be less than or at least equal to j. Thus all we need to do is to replace n_j in all the equations above by

$$n_{j_p} = n_j\frac{\xi j}{L}. \tag{17}$$

4 Performance Evaluation

Table 1 shows the performance of HARPS, RPS and LM for broadcast encryption (both by TA and by peers) for 3 different values of the group sizes (roughly a thousand, million and billion) for $k = 500$. For LM and HARPS, $L = 64$ (the largest hash depth permissible). For HARPS and RPS we assume that an optimal value of $\xi = \frac{k}{P}$ is chosen for each case ((for a given r and G) - the optimal value of ξ is also indicated within parenthesis for the case of $G = 1024$.

Some of the points worth noting are the following:

المنارة للاستشارات
www.manaraa.com

Table 1: Performance of broadcast encryption in terms of number of encryptions necessary *per revoked node*) for various values of r using random KPSs, for group sizes $G = 2^{10}, 2^{20}, 2^{30}$ (roughly a thousand, million and billion), for $k = 500$ and $L = 512$. For RPS and HARPS the corresponding optimal choice of ξ, are also indicated within parenthesis.

Broadcast Encryption By TA									
r	$G = 2^{10}$			$G = 2^{20}$			$G = 2^{30}$		
	HARPS	RPS	LM	HARPS	RPS	LM	HARPS	RPS	LM
2	1.456(0.994)	1.710(0.926)	1.457	2.886	3.548	2.888	4.504	5.638	4.511
4	1.456(0.867)	1.687(0.700)	1.485	3.065	3.666	3.173	4.924	6.010	5.168
8	1.393(0.593)	1.586(0.442)	1.528	3.102	3.644	3.605	5.095	6.098	6.247
32	0.139(0.189)	1.272(0.137)	1.645	2.887	3.333	5.873	4.951	5.825	21.56
64	0.988(0.099)	1.096(0.073)	1.714	2.721	3.128	17.667	4.777	5.600	1.1e4
128	0.829(0.052)	0.912(0.039)	1.749	2.544	2.914	267.07	4.580	5.355	2.7e5
Broadcast Encryption By Peers									
2	1.464(0.996)	1.738(0.923)	1.469	2.915	3.625	2.9174	4.571	5.910	4.579
4	1.521(0.862)	1.833(0.667)	1.558	3.299	4.248	3.455	5.453	7.459	5.865
8	1.596(0.667)	1.961(0.368)	1.863	3.894	5.348	5.898	7.017	11.452	315

Table 2: Performance of HARPS and RPS designed for $r = 64$, $G = 2^{20}$, for other values of r. For HARPS, $P = 6080, k = 500, L = 512$. For RPS $P = 8435, k = 500$. The figure represent the *total* number of encryptions needed to revoke r nodes.

r	1	2	16	32	64	72	128
n_H	145	145	145	145	174	195	375
n_R	197	197	197	197	203	1972	8.5e5

1. For small group sizes the efficiency of BE using RKPSs is significantly better than tree-based methods (the most efficient of which requires about 1.25 encryptions per revoked node.
2. The same keys can be used for various group sizes with increasing efficiency as group size reduces. For tree-based schemes the number of secrets provided to each node would depend on the *maximum* possible group size. Even though the same secrets can be used for smaller group sizes, they cannot be used with greater efficiency. In the example above we have used the same $k = 500$ for all group sizes.

However, the efficiency of BE with RKPSs depends on the size of r. For HARPS and RPS, for each r we have chosen the optimal value of ξ (the optimal value of ξ also depends on the group size G, but to a much smaller extent). For LM (by definition), ξ is always one. As it is not practical to change the value of ξ after deployment, for any deployment of RPS / HARPS (with some k, ξ), there would be an *optimal* choice of r.

For instance, consider a HARPS deployment optimized for $r = 64$ with $k = 500, P = 6080, L = 64$, for a group size of 1 million. While the number of revocations per node is only about 2.721 encryptions per node if $r = 64$, in practice smaller sizes of r need to be catered for. In general, HARPS or RPS designed for large r would perform very poorly for small r. Table 2 lists the number of encryptions needed for revoking r nodes for $r = 1, 2, 4, 8, 16, 32, 64$ and 128 (in terms of *total number of encryptions* - not number of encryptions per revoked node). Note that the *same* number of encryptions are needed for revoking 1 to 32 nodes! This is due to the fact that a minimum number of encryptions (145 for HARPS and 197 for RPS in this case) have to be transmitted to achieve the desired probability of outage - irrespective of the number of revoked nodes. The loss in efficiency is very high especially for small r. However, HARPS designed for some r degrades much more gracefully for larger r when compared to RPS.

While the LM scheme does not have this problem, it does not perform very well for large r (especially for large group sizes). It might appear at first sight that revocation can always be performed in smaller *batches* using LM (where a subset of nodes is revoked in each batch) with small batch sizes, without any loss of efficiency[1]. However, this is not desirable. While the tree-based schemes by Noar et al [3] and Halevy et al [4], are resistant to collusion of all revoked nodes *even if they were revoked in different "batches"*, for broadcast encryption using random KPSs, the system is resistant only to collusion of *all nodes in a batch*.

Thus a solution to this problem may be to use many systems in parallel with different values of ξ (in practice 2 systems with $xi = 1$ and $\xi = 0.1$ may be sufficient) could be used. For example, we could use $k_1 = k_2 = k_3 = k_4 = k_5 = 200$, and $P_1 = 200, P_2 = 400, P_3 = 800, P_4 = 1600, P_5 = 3200$), or effectively $k = 400 \times 5 = 2000$ and $P = 400 + 800 + 1600 + 3200 + 6400 = 12400$). Alternately a system could have P keys $K_1 \cdots K_P$, and the probability that key i is assigned to any node could be $\xi_i = \mu f(i)$ where μ is appropriately chosen such that $0 < \mu f(i) \leq 1$ and $f(i)$ is a monotonic function of i. Figure 1 (left) plots the bandwidth required vs r (from 1 to 200) for such a hybrid HARPS deployment with $k = 200 \times 5 = 1000$, and $P = 200 + 250 + 800 + 1600 + 3200 = 6050$, and $L = 512$, for group sizes of roughly a thousand, million and a billion. There is however a *soft* upper bound on the size of r (which also depends on the group size G). As r increases, it may not be possible for the TA to find enough keys to use for encrypting the broadcast,

[1] The results in Table 1 are in terms of number of encryptions *per* revoked node.

المنارة للاستشارات
www.manaraa.com

and thus will not be able to attain a low enough p_o to ensure that all intended nodes can be reached.

Figure 1 (right) is the plots for broadcast encryption by peers, for group sizes of a thousand and a million Broadcast encryption by peers is impractical for large group sizes and r - the number of keys a source can employ is substantially less than the number of keys the TA can.

4.1 Overheads

Apart from the encryptions of the secrets, for BE schemes the source should also indicate the identities of the revoked nodes. For revocation using random KPSs (while this can also be done), it is more efficient (both in terms of bandwidth needed and computational complexity at the receiver) to instead provide the *indexes of the keys used* for encryption (and additionally, their corresponding hash depths in case of HARPS and LM).

The expression for the overheads can be obtained easily by considering the entropies of the indexes and the hash depths. Let o_I be the entropy of overheads for transmitting indexes of the subset of P keys that are used for encrypting the broadcast secret (n_e out of P keys are used). If we define $\mathbb{H}(x) = -x\log_2(x)$, we have

$$o_I = P\left\{\mathbb{H}\left(\frac{n_e}{P}\right) + \mathbb{H}\left(\frac{P-n_e}{P}\right)\right\} \text{ bits.}$$

Let O_d be the entropy of the hash depths for the n_e encryption keys used. Or

$$o_d = n_e \sum_{j=q}^{L} \mathbb{H}\left(\frac{n_j}{n_e}\right) \text{ bits.}$$

Typically, the overheads range between 10 to 40 bits per revoked node. However, more important than the reduced overhead (compared to tree-based schemes) is the fact that this caters for privacy - *by protecting the identities of the revoked nodes*.

4.2 Broadcast Authentication

In practice, broadcasts meant for revoking nodes need to be authenticated. This calls for the ability to cater for broadcast authentication. Various techniques for broadcast authentication (without employing asymmetric cryptography[2]) have been considered in the literature. Such techniques can be divided into two main classes:

1. Instantaneous authentication techniques
2. Authentication using delayed disclosure

With the first class of methods (based on key pre-distribution), broadcast authentication can be achieved by appending many key based message authentication codes (MAC) [27], [26] - [28], - one corresponding to each of the k keys the source node it has in its key ring (all random KPSs cater for broadcast authentication). Any verifier may be able to verify a subset of the MACs. However, such approaches have the disadvantage of large bandwidth requirements (the number of appended authentication codes, k may be high).

The second class of methods, based on one-way hash chains [31] have been investigated by various researchers [32] - [34]. With this approach, the source creates a one-way hash chain and uses a value from the hash chain to calculate the message authentication code for a message to be authenticated. Later the pre-image of the value used is made public. At this point the verifiers are assured that the source that transmitted the first message was the one who released the pre-image (as no one else can, in practice compute the pre-image of a disclosed value). However, such methods typically need to be *bootstrapped* from a pre-authenticated value[3].

Thus a satisfactory solution may be to use RKPSs for bootstrapping hash-chain based techniques. In other words, the more bandwidth expensive authentication using RKPSs could be used for the first broadcast to authenticate the "commitment" key in the hash chain, and subsequent broadcasts can be authenticated by releasing pre-images of the commitment.

4.3 Pros and Cons

A summary of advantages and disadvantages of broadcast encryption using RKPSs (as opposed to state-of-the-art tree-based schemes in [3] and [4]) are as follows

1. Advantages:
   (a) broadcast by peers without the use of asymmetric cryptography
   (b) higher efficiency for smaller group sizes
   (c) flexible group / universe size - no hard limit on the maximum size of groups or the total number of nodes.
   (d) ability to protect identity of revoked nodes.
   (e) lower bandwidth for overheads
   (f) the secrets used for broadcast encryption can also be used for mutual authentication and broadcast authentication.
2. Disadvantages:
   (a) Limit (though soft) on the size of total number of nodes that can be revoked while still catering for resistance against collusion of *all* revoked nodes.
   (b) For very large group sizes (say a billion or above) it may be impractical for the source of the broadcast to *verify* if *all* privileged nodes can decipher the broadcast. Thus
      i. the source may have to sacrifice the bandwidth efficiency (by using more keys for encrypting the broadcast secret) and target a value of outage probability p_o that is considerably lower than the inverse of the group size ($p_o << \frac{1}{G}$), in order to render the probability of such an event very low, *or*

[2] If asymmetric cryptography is feasible, broadcast authentication can be achieved using digital signatures which can be verified by any receiver.

[3] The first class of methods - methods based on key pre-distribution - do not need to be bootstrapped.

المنارة للاستشارات
www.manaraa.com

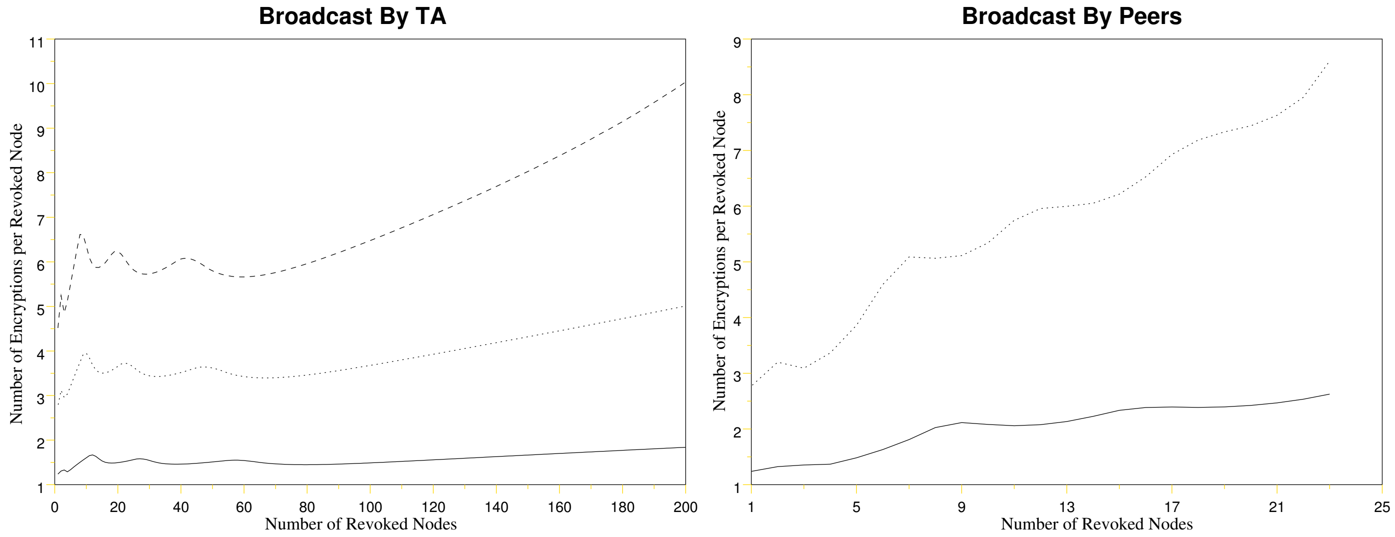

Figure 1: Performance of broadcast encryption using a hybrid HARPS scheme employing $k = 5 \times 200 = 1000$, $P = 200 + 250 + 800 + 1600 + 3200 = 6050$, and $L = 512$, for broadcasts by TA (left) and by peers (right). Dashed line (only in the left figure) is for $G = 2^{30}$. Dotted lines and unbroken lines (in both figures) are for $G = 2^{20}$ and $G = 2^{10}$ thousand respectively.

ii. the system should cater for such "accidentally" missed nodes to approach the source and receive the secret through alternate channels (the second approach is perhaps more practical for large group sizes).

4.4 Potential Applications

4.4.1 Wireless Ad Hoc Network Security

Security solutions for (wireless) mobile ad hoc networks could benefit greatly if nodes in a vicinity could establish shared secrets while *selectively excluding* some nodes. As an example, it could be very useful in many ad hoc routing protocols if nodes could establish a shared secret with all their 2-hop neighbors - which are not provided to 1-hop neighbors [35], [36]. This could for instance, help nodes authenticate their messages to their 2-hop neighbors and ensure that one-hop neighbors do not modify the contents of the packet they forward. This could be easily realized by a broadcast which "revokes" all one hop neighbors.

In such scenarios, it is more important that the explicitly revoked nodes are *denied* access to the secret, than ensuring that all other nodes actually receive the secret. Obviously, the set of excluded nodes need not be based just on hop counts - there may be other reasons to choose the set of revoked nodes. For instance, in scenarios where nodes maintain a "neighborhood watch" [37] and rate the trust-worthiness of each node, nodes may need to exchange information, while shielding it from the node (or nodes) with "questionable morals".

4.4.2 Publish-Subscribe Systems

In publish-subscribe (pub-sub) [38] systems broadcast encryption will be very useful for distributing secrets to a set of subscribers (or more specifically revoking a set of current subscribers by providing continuing subscribers with a new secret). With the possibility of broadcast encryption by peers, any node has the ability to become a publisher, and distribute secrets to other nodes. In pub-sub systems one very important issue is protection of the privacy of publisher-subscriber relationships. Thus tree-based broadcast encryption schemes (even when they are used with asymmetric cryptographic primitives to facilitate broadcasts by peers) which need to explicitly specify the identities of the revoked nodes may not be very suitable for this purpose.

5 Conclusions

We discuss the applicability of random key pre-distribution schemes for broadcast encryption and argue that this may be a useful paradigm in many application scenarios. One of the main shortcoming of broadcast encryption using random KPSs is the limit on the number of nodes that can be revoked. While BE can still be performed in batches to overcome this "soft" limit, the collusion resistance holds only for nodes within each batch.

However, in scenarios where privacy is an important issue, and / or it may be infeasible to employ asymmetric cryptographic primitives, BE using random KPSs can be a very useful tool - especially since the same secrets used for BE can also be used for mutual authentication and broadcast authentication.

References

[1] A. Fiat, M. Noar, "Broadcast Encryption," Lecture Notes in Computer Science, Advances in Cryptology, Springer-Verlag, **773**, pp 480–491, 1994.

المنارة للاستشارات
www.manaraa.com

[2] R. Canetti, T. Malkin, K. Nissin, “Efficient Communication-Storage Tradeoffs for Multicast Encryption,” EUROCRYPT 1999, pp 459–474.

[3] D. Noar, M. Noar, J. Lotspiech, “Revocation and Tracing Routines for Stateless Receivers,” Lecture Notes in Computer Science, Advances in Cryptology, Springer-Verlag, **2139**, 2001.

[4] D. Halevy, A. Shamir, “The LSD Broadcast Encryption Scheme,” Advances in Cryptology - CRYPTO 2002: 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002.

[5] C.K. Wong, M. Gouda, S. Lam, “Secure Group Communications using Key Graphs,” Proceedings of SIGCOMM 98, pp 68–79, 1998.

[6] J. Lotspiech, S. Nusser, F. Pestonoi, “Anonymous Trust: Digital Rights Management using Broadcast Encryption,” Proceedings of the IEEE, **92** (6), pp 898–909, 2004.

[7] Eli Gafni A1, Jessica Staddon A2, Yiqun Lisa Yin, “Efficient Methods for Integrating Traceability and Broadcast Encryption,” Advances in Cryptology - CRYPTO’99: 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 1999.

[8] G. Kreitz, “Optimization of Broadcast Encryption Schemes,” Master’s Thesis, Royal Institute of Technology, Sweden, Feb 2005.

[9] D. R. Stinson, “On Some Methods for Unconditionally Secure Key Distribution and Broadcast Encryption,” Designs, Codes and Cryptography, Kluwer Academic Publishers Norwell, MA, USA, 1997.

[10] B. C. Neuman, T. Ts’o, “Kerberos: An Authentication Service for Computer Networks”, IEEE Communications, **32**(9), pp 33–38. September 1994.

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

[12] S.Kiran, P. Lareau, S.Lloyd, “PKI Basics - A Technical Perspective,” PKI-Forum (http://www.pkiforum.org), November 2002.

[13] J. Crowcroft, “Scalable and Ubiquitous Computing Systems,” Grand Challenges in Computing (Research), edited by T. Hoare and R. Milner, 2004.

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

[15] R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, T. Rabin, “Tamper Proof Security: Theoretical Foundations for Security Against Hardware Tampering,” Theory of Cryptography Conference, Cambridge, MA, February 2004.

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

[17] 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.

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

[19] R. Di Pietro, L. V. Mancini, A. Mei, “Random Key Assignment for Secure Wireless Sensor Networks,” 2003 ACM Workshop on Security of Ad Hoc and Sensor Networks, October 2003.

[20] S. Zhu, S. Xu, S. Setia S. Jajodia, “Establishing Pair-wise Keys For Secure Communication in Ad Hoc Networks: A Probabilistic Approach,” Proc. of the 11th IEEE International Conference on Network Protocols (ICNP’03), Atlanta, Georgia, November 4-7, 2003.

[21] W. Du, J. Deng, Y.S. Han. P.K.Varshney, “A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks,” Proceedings of the 10th ACM Conference on Computer and Communication Security, pp 42–51, 2003.

[22] M. Ramkumar, N. Memon, R. Simha, “Pre-Loaded Key Based Multicast and Broadcast Authentication in Mobile Ad-Hoc Networks,” Globecom-2003.

[23] D. Liu, P.Ning, “Establishing Pairwise Keys in Distributed Sensor Networks,” Proceedings of the 10th ACM Conference on Computer and Communication Security, Washington DC, 2003.

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

[25] 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.

[26] N. Alon, “Probabilistic Methods in External Finite Set Theory,” in *Extremal Problems for Finite Sets,*, pp 39-57, 1991.

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

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

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

المنارة للاستشارات

www.manaraa.com

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

[31] L. Lamport, “Password Authentication with Insecure Communication,” Communications of the ACM, 24(11):770-772, November 1981.

[32] S. Cheung, ”An Efficient Message Authentication Scheme for Link State Routing”, Proceedings of the 13th Annual Computer Security Applications Conference, San Diego, California, December 1997, pp. 90-98.

[33] R.J. Anderson, F. Bergadano, B. Crispo, J.H. Lee, C. Manifavas and R.M. Needham, “ A New Family of Authentication Protocols,” ACM Operating Systems Review, vol. 32, n. 4, pp. 9-20, October 1998, ACM Press.

[34] A. Perrig, R. Canetti, D. Song, D. Tygar, “Efficient and Secure Source Authentication for Multicast,” in Network and Distributed System Security Symposium, NDSS ’01, Feb. 2001.

[35] P-W Yau, C. J. Mitchell, “2HARP: A Secure Routing Protocol for Mobile Ad Hoc Networks,” 5th World Wireless Congress (WWC 2004), San Francisco, USA, May 2004,

[36] X. Du, Y. Wang, J. Ge, Y. Wang, “A Method for Security Enhancements in AODV Protocol,” 17 th International Conference on Advanced Information Networking and Applications (AINA), Xian, China, 2003.

[37] S. Buchegger, J-Y. Le Boudec, “Nodes Bearing Grudges: Towards Routing Security, Fairness and Robustness in Mobile Ad Hoc Networks,” Proceedings of Tenth Euromicro Workshop on Parallel Distributed and Network-based Processing, 2002.

[38] P.T. Eugster, P.A. Felber, R. Guerraoui, A-M. Kermarrec, “The Many Faces of Publish/Subscribe,” Technical Report, URL citeseer.ist.psu.edu/649723.html.

[39] C. Wang, A. Carzaniga, D. Evans, A. L. Wolf, “Security Issues and Requirements for Internet-Scale Publish-Subscribe Systems,” Hawaii International Conference on System Sciences, January, 2002.

المنارة للاستشارات

www.manaraa.com