A new key-predistribution scheme for highly mobile sensor networks

Abhijit Das[*] Bimal Kumar Roy[†]

[*]Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur 721 302, India
abhij@cse.iitkgp.ernet.in

[†]Applied Statistics Unit
Indian Statistical Institute, Calcutta 700 035, India
bimal@isical.ac.in

Abstract. In this paper, we propose a new deterministic key predistribution scheme for secure communication in wireless sensor networks. Our scheme provides better trade-off among scalability, computation overhead, connectivity and resilience against node captures than existing key predistribution schemes, particularly in situations where the nodes in the network are highly mobile.

1 Introduction

Secure communication in a network of resource-constrained sensor nodes has been an important area of research in the recent past. In a network where the nodes are static (or have only very limited mobility), deployed nodes establish secure links with neighboring nodes using predistributed keys by a process called shared key discovery. In this paper, we address the situation where the nodes in the sensor network are highly mobile. The nodes may reestablish secure links by carrying out the shared key discovery phase periodically. The resulting overhead becomes unacceptably high under most of the existing schemes. In view of this, we look for a key predistribution scheme with modified requirements. We assume that the nodes are aware of their movements. If not, a node may periodically probe its neighborhood and invokes a key reestablishment procedure if it finds a significant change in its neighborhood. Suppose that a node u attempts to reestablish keys in a new neighborhood, and v is an arbitrary neighbor of u. The reestablishment process should meet the following desirable properties.

- The reestablishment process is initiated by u and requires the cooperation of each neighboring node v.
- The node u is allowed to perform some reasonable amount of computation. If m is the number of nodes in the neighborhood of u, then a computation of amount $\mathrm{O}(m)$ is the absolute minimum.
- Each neighboring node v must not be subject to too much computation and/or communication overhead. A good algorithm corresponds to this overhead to be of an amount $\mathrm{O}(1)$ per neighbor.

The basic scheme [5] and its variants [3], the polynomial-pool scheme [7] and the matrix-based scheme [4] incur sufficiently more overhead than this desirable minimum. Several deterministic and hybrid schemes [1, 2, 6, 10] are available in the literature. These schemes do not make it clear how u can quickly compute the key ring of v, often without invoking the entire key predistribution procedure. The scheme of [8] addresses the issue of the essential asymmetry in the key reestablishment phase as mentioned above. However, the computation overhead of u in this scheme is very high. Ruj and Roy propose a scheme [9] which seems to be the most appropriate for mobile networks, but has small maximum supported network sizes and poor resilience against node captures.

In this paper, we improve Ruj and Roy's scheme and obtain a trade-off among scalability, security and connectivity, which appears acceptable in most practical applications. Our scheme achieves the lower bounds on computation overhead for both u and v. A matrix-layout based scheme (henceforth referred to as the ML scheme) is at the heart of our modification (Section 2). The ML scheme achieves high connectivity, but suffers from somewhat poor resilience, particularly against selective node captures (Sections 3 and 4).

2 Our matrix-layout based (ML) scheme

2.1 Construction of the layout matrix

Let t denote the maximum number of symmetric keys that each sensor node can store in its memory. We assume that t is even. We take $d = t + 2$ and construct a $d \times d$ matrix as follows. We first fill the main diagonal of the matrix by a special symbol which does not stand for the id of any key or node. We then fill out the triangular region below (and excluding) the main diagonal and above (and including) the reverse main diagonal in the column-major order by the integers $1, 2, 3, \ldots, \left(\frac{t+2}{2}\right)^2$. Subsequently, we fill out the triangular region above (and excluding) the main diagonal and above (and including) the reverse main diagonal in the row-major order by the integers $\left(\frac{t+2}{2}\right)^2 + 1, \left(\frac{t+2}{2}\right)^2 + 2, \ldots, 2\left(\frac{t+2}{2}\right)^2$. Finally, we reflect about the reverse main diagonal the entries above this diagonal in order to fill the region below this diagonal, and make the matrix symmetric with respect to the reverse main diagonal. For example, for $t = 4$, the layout matrix is constructed as follows.

$$\begin{pmatrix} \ast & & & & & \\ 1 & \ast & & & & \\ 2 & 6 & \ast & & & \\ 3 & 7 & 9 & \ast & & \\ 4 & 8 & & & \ast & \\ 5 & & & & & \ast \end{pmatrix} \rightarrow \begin{pmatrix} \ast & 10 & 11 & 12 & 13 & 14 \\ 1 & \ast & 15 & 16 & 17 & \\ 2 & 6 & \ast & 18 & & \\ 3 & 7 & 9 & \ast & & \\ 4 & 8 & & & \ast & \\ 5 & & & & & \ast \end{pmatrix} \rightarrow \begin{pmatrix} \ast & 10 & 11 & 12 & 13 & 14 \\ 1 & \ast & 15 & 16 & 17 & 13 \\ 2 & 6 & \ast & 18 & 16 & 12 \\ 3 & 7 & 9 & \ast & 15 & 11 \\ 4 & 8 & 7 & 6 & \ast & 10 \\ 5 & 4 & 3 & 2 & 1 & \ast \end{pmatrix}$$

For easy future references, we call the triangular region in the layout matrix storing $1, 2, \ldots, \left(\frac{t+2}{2}\right)^2$ as the left triangle (LT), and the triangular region storing $\left(\frac{t+2}{2}\right)^2 + 1, \left(\frac{t+2}{2}\right)^2 + 2, \ldots, 2\left(\frac{t+2}{2}\right)^2$ as the top triangle (TT). Their reflections about the reverse main diagonal are respectively called the bottom triangle (BT) and the right triangle (RT).

We assume that matrix indexing is zero-based. Thus, the top-left element of the layout matrix has index $(0, 0)$, and the bottom-right element has index $(t + 1, t + 1)$. The following formula converts an index (i, j) to the entry of the layout matrix at the (i, j)-th location.

$$f(i,j) = \begin{cases} * & \text{if } i = j, \\ -j^2 + (t+1)j + i & \text{if } i > j \text{ and } i \leq t+1-j, \\ f(t+1-j, t+1-i) & \text{if } i > j \text{ and } i > t+1-j, \\ \left(\frac{t+2}{2}\right)^2 + f(j,i) & \text{if } i < j. \end{cases}$$

Given (i, j), one can compute $f(i, j)$ in O(1) time (using a few single-precision operations only).

2.2 Key predistribution

Before deployment of the sensor nodes, the key-ring of each node is loaded with t symmetric keys (like AES keys). These keys are selected from a pool of $T = 2\left(\frac{t+2}{2}\right)^2$ randomly chosen keys having the ids $1, 2, 3, \ldots, T$. The maximum number of nodes supported by our scheme is also $N = 2\left(\frac{t+2}{2}\right)^2$. The nodes in the network are also given ids in the range $1, 2, 3, \ldots, N$.

For each position (i, j) in the triangle LT in the layout matrix, one first computes $u = f(i, j)$. All the entries in the j-th column in the matrix are then considered, except u itself and the special symbol *. The t keys with ids equal to these t elements are loaded in the key-ring of u along with the respective key ids. In addition, the location (i, j) is also stored in the memory of u.

Subsequently, for each position (i, j) in the triangle TT, one computes $u = f(i, j) = \left(\frac{t+2}{2}\right)^2 + f(j, i)$. The key-ring of u is loaded with the keys whose ids are the elements of the i-th row of the matrix (except u and *).

2.3 Shared key discovery

Suppose that a node u wants to establish a key with a node v. Let (u_i, u_j) and (v_i, v_j) denote the locations of the nodes u and v in the layout matrix.

Assume that u is located in the triangle LT in the layout matrix. Figures 1 and 2 explain this situation. First consider the case that (v_i, v_j) too is in the triangle LT. If $u_j = v_j$, then u and v share the $t - 1$ keys with ids $f(i, u_j)$ for $i \neq u_i, v_i, u_j$ (Figure 1(c)).

If u and v are both in the triangle LT and $u_j \neq v_j$, we have a situation described in Figure 1(a). By construction, the v_j-th column of the layout matrix is identical to its $(t + 1 - v_j)$-th row. The u_j-th column and the $(t + 1 - v_j)$-th row intersect at the unique location $(t + 1 - v_j, u_j)$. If this location is distinct from (u_i, u_j), then u and v share the unique key with id $f(t + 1 - v_j, u_j)$. If, on the other hand, $u_i = t + 1 - v_j$, then u and v do not share a key (Figure 2(a)).

Finally, suppose that (v_i, v_j) is in the triangle TT of the layout matrix. In this case, u and v share the unique key with id $f(v_i, u_j)$ (Figure 1(b)), unless $v_i = u_i$ (Figure 2(b)) or $v_i = u_j$ (Figure 2(c)).

Fig. 1. Two nodes u, v sharing keys

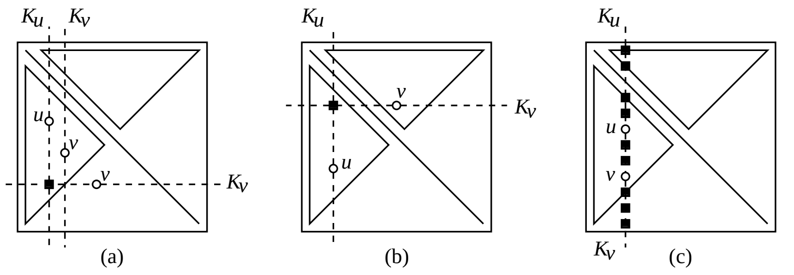
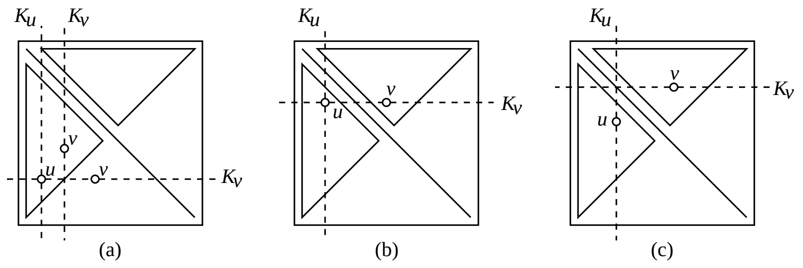


Fig. 2. Two nodes u, v not sharing keys



To sum up, the node u identifies in O(1) time a key shared with v (provided that such a key exists). If m denotes the number of neighbors of u in its new neighborhood, then using only O(m) computation and communication overhead u can reestablish its key connectivity records, whereas each new neighbor of u incurs only O(1) computation and communication overhead. No key predistribution schemes proposed earlier achieve such a high efficiency.

3 Analysis of the ML scheme

3.1 Connectivity

Figure 2 shows the three situations in which two nodes u and v do not share a key. The number of such ordered pairs (u, v) is $(t+1)(t+2)(2t+3)/3 = \Theta(t^3)$. The total number of ordered pairs (u, v) is $N(N-1) = \Theta(t^4)$, where $N = 2\left(\frac{t+2}{2}\right)^2$.

Proposition The connectivity of the ML scheme is $p = 1 - \Theta\left(\frac{1}{t}\right)$. More precisely, $p \approx 1 - \frac{8}{3t}$ except for very small values of t. In particular, p is very close to 1 for all practical values of t. Moreover, the number of ordered pairs of nodes sharing $t-1$ common keys is $t(t+2)(2t+5)/6 = \Theta(t^3)$, i.e., most pairs of connected nodes share unique keys.

Table 1. Comparison of parameters for several schemes ($t = 100$)

	Key-pool	Max net-	Connec-	Simulated values of $C(s)$ for $s =$						
Scheme	size	work size	tivity	10	20	50	100	150	200	250
Basic	5202	–	0.859	0.176	0.322	0.621	0.856	0.946	0.979	0.992
	2922	–	0.971	0.294	0.502	0.825	0.969	0.995	0.999	1.000
	2832	–	0.974	0.302	0.513	0.834	0.973	0.995	0.999	1.000
Ruj & Roy I	1326	1326	1.0	0.166	0.470	0.918	0.998	1.000	1.000	1.000
Ruj & Roy II	1326	2652	1.0	0.113	0.393	0.898	0.998	1.000	1.000	1.000
ML I	5202	5202	0.974	0.236	0.414	0.702	0.898	0.964	0.989	0.993
ML II	5202	10404	0.971	0.176	0.324	0.612	0.863	0.945	0.979	0.991

3.2 Security analysis

The computational efficiency and high connectivity of the ML scheme come at a cost. The resilience of the network against node captures is somewhat poor. Capturing only $t + 2$ nodes (one node from each column of the triangle LT and one node from each row of the triangle TT) reveals all the keys to an adversary. However, if we adopt a model of random node capture, we get a resilience similar to the basic scheme [5] under the assumption that the deterministically distributed keys behave as randomly distributed keys. This assumption is, however, not very accurate, and we obtain a resilience slightly smaller than that of the basic scheme with the same pool size.

3.3 Doubling the maximum supported network size

The key-rings of the nodes in the triangle LT (resp. TT) are based on the columns (resp. rows) of the matrix. We now distribute the same keys to a new set of $N = 2\left(\frac{t+2}{2}\right)^2$ nodes. In this case the key rings of the nodes in the triangle LT (resp. TT) are based on the rows (resp. columns) of the layout matrix. The connectivity among the new nodes remains identical with that among the old nodes. The cross connectivity among the old nodes and the new nodes continues to remain $1 - \Theta(1/t)$. The maximum supported network size now becomes $4\left(\frac{t+2}{2}\right)^2 \approx t^2$. This extended scheme is called the ML Scheme II.

4 Comparison with other schemes

In Table 1, we compare our schemes with the basic scheme [5] and Ruj and Roy's schemes [9]. We take the capacity of the key-ring in each sensor node to be $t = 100$ keys. By $C(s)$, we denote the fraction of compromised links among uncaptured nodes, when s randomly chosen nodes are captured.

Let us first compare our scheme with Ruj and Roy's scheme. By reducing the amount of overlap between key rings, we have increased both the key-pool size and the maximum supported network size by a factor of nearly 4. This gain comes at the cost of some marginal loss of connectivity. For $t = 100$, the loss of connectivity is less than 3%. When the number of captured nodes is quite small, Ruj and Roy's schemes provide better resilience than our schemes because of larger overlaps of key rings of the nodes. However, as the number of captured nodes increases, a smaller key-pool size in Ruj and Roy's schemes makes their resilience noticeably poorer than that of ML schemes.

The basic scheme supports networks of any size. When we use the same key-pool size as the ML schemes, the basic scheme yields poorer connectivity but higher resilience against node captures. On the other hand, if we reduce the key-pool size for the basic scheme so as to achieve the same connectivity as the ML schemes, its resilience becomes poorer than that for the ML schemes.

5 Conclusion

In this paper, we propose the matrix-layout (ML) scheme for deterministic key predistribution in a sensor network. Our scheme is very suitable for mobile networks, since it optimizes the computation and communication overhead of key reestablishment. However, poor resilience of our scheme against node captures (particularly selective captures) is expected to attract further research attention. Several ad hoc techniques (like using multiple copies of the ML scheme) can increase resilience at the cost of decreased connectivity.

References

1. S. A. Camtepe and B. Yener, 'Combinatorial design of key distribution mechanisms for wireless sensor networks', ESORICS 2004, LNCS 3193, 293–308, 2004.
2. D. Chakrabarti, S. Maitra and B. K. Roy, 'A key pre-distribution scheme for wireless sensor networks: merging blocks in combinatorial design', Intl. Jl. Inf. Sec. 5(2), 105–114, 2006.
3. H. Chan, A. Perrig and D. Song, 'Random key predistribution for sensor networks', IEEE Symposium on Security and Privacy, 197–213, 2003.
4. W. Du, J. Deng, Y. S. Han and P. K. Varshney, 'A pairwise key pre-distribution scheme for wireless sensor networks', CCS'03, 42–51, 2003.
5. L. Eschenauer and V. D. Gligor, 'A key management scheme for distributed sensor networks', CCS'02, 41–47, 2002.
6. J. Lee and D. R. Stinson, 'Deterministic key predistribution schemes for distributed sensor networks', SAC 2004, LNCS 3357, 294–307, 2005.
7. D. Liu and P. Ning, 'Establishing pairwise keys in distributed sensor networks', CCS'03, 52–61, 2003.
8. M. Mehta, D. Huang and L. Harn, 'RINK-RKP: A scheme for key predistribution and shared-key discovery in sensor networks', IPCCC'05, 193–197, 2005.
9. S. Ruj and B. Roy, 'Key predistribution using partially balanced designs in wireless sensor networks', ISPA 2007, LNCS 4742, 431–445, 2007.
10. D. S. Sánchez and H. Baldus, 'A deterministic pairwise key pre-distribution scheme for mobile sensor networks', Securecomm'05, 277–288, 2005.