

**A Combinatorial Approach to Key Predistribution
for Distributed Sensor Networks**

Douglas R. Stinson

**David R. Cheriton School of Computer Science
University of Waterloo**

and

Jooyoung Lee

**National Security Research Institute
Korea**

Distributed Sensor Networks

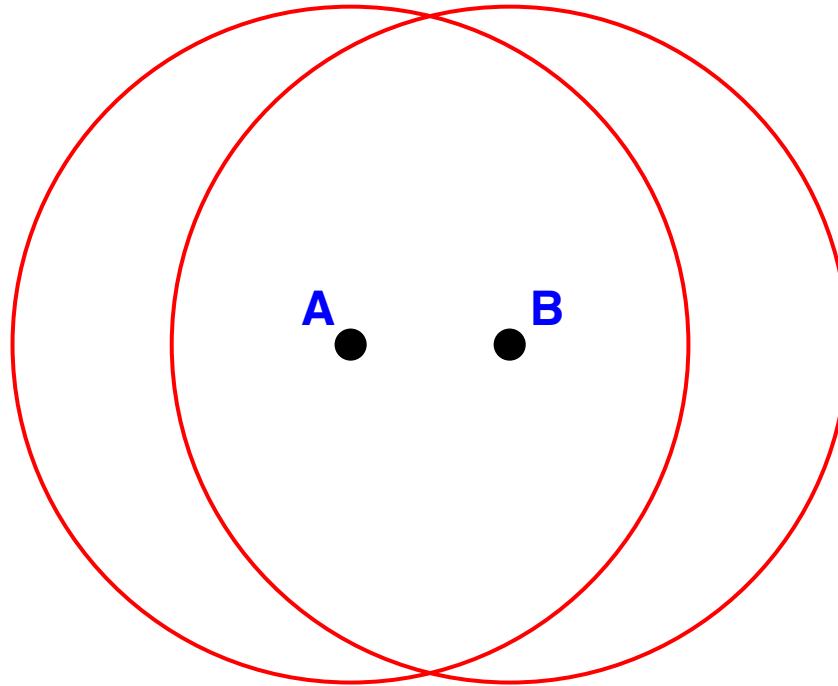
- **sensor nodes** have limited computation and communication capabilities
- a network of 1000 – 10000 sensor nodes is distributed in a random way in a possibly hostile physical environment
- the sensor nodes operate unattended for extended periods of time
- the sensor nodes have no external power supply, so they should consume as little battery power as possible
- usually, the sensor nodes communicate using secret key cryptography
- a set of secret keys is installed in each node, before the sensor nodes are deployed, using a suitable **key predistribution scheme** (or KPS)
- nodes may become inactive or they may be stolen by an adversary (this is called **node compromise**)

Fundamental Algorithms for DSNs

Eschenauer and Gligor (2002) studied the following problems for DSNs:

- **key predistribution** How do we assign keys to sensor nodes? We do not want to use a single key across the whole network due to the possibility of node compromise. So each node will receive a moderate sized **key ring**.
- **shared-key discovery** Two nodes can communicate directly only if they are in close physical proximity **and** they have a common key. We need an efficient method to determine if two nodes share a common key.
- **path-key establishment** Nodes that cannot communicate directly should be able to communicate via a **multi-hop path**. We need an efficient method for two nodes to determine a secure multi-hop path. (We focus on **two-hop paths**.)

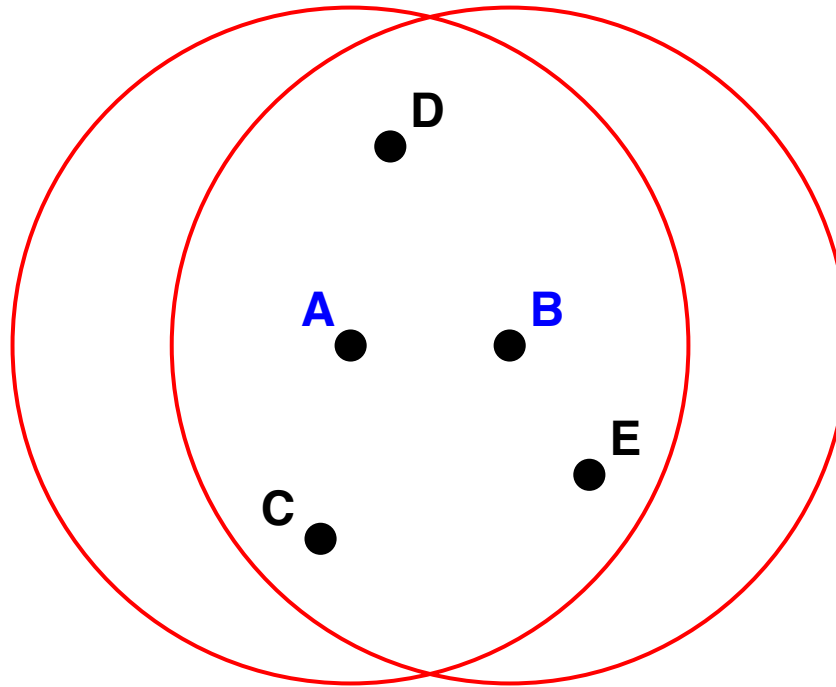
Shared-key Discovery



A has keys k1, k3, k5

B has keys k2, k4, k6

Path-key Establishment (1)



A has keys k1, k3, k5

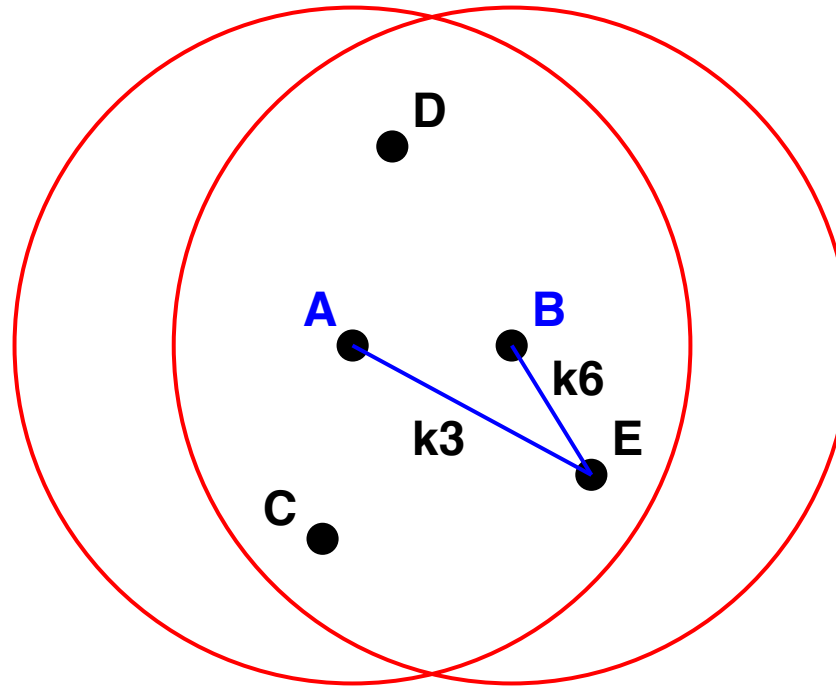
B has keys k2, k4, k6

C has keys k1, k3, k7

D has keys k2, k6, k7

E has keys k3, k6, k7

Path-key Establishment (2)



A has keys k1, k3, k5

B has keys k2, k4, k6

C has keys k1, k3, k7

D has keys k2, k6, k7

E has keys k3, k6, k7

Key Predistribution Schemes for DSNs

A key predistribution scheme for a DSN must balance several factors:

1. the number of keys / node, say k , should be relatively small (due to **memory constraints**); k is usually between 50 and 200
2. each key should be assigned to the same number of nodes, say r (to minimize the effect of a **node compromise**)
3. each node should have a common key with “many” other nodes (to equalize and maximize **network connectivity**)
4. if two nodes N_i, N_j do not have a common key, there should exist “many” nodes N_h such that N_i and N_h have a common key, and N_h and N_j have a (different) common key (to ensure that **two-hop paths** can be found easily)

Randomized KPS

- Eschenauer and Gligor (2002) suggested using **random** KPSs for DSNs
- a **key pool** consisting of v random keys is constructed
- each of the b nodes in the network is given a random subset of k keys chosen from the key pool
- on average, every key is given to r nodes, where $r = vk/b$
- the Eschenauer/Gligor scheme has good behaviour with respect to connectivity and resilience, but shared-key discovery is inefficient
- several variations of the Eschenauer/Gligor schemes have been proposed, but most of these variations are also randomized schemes

Deterministic KPS

- Çamtepe and Yener (2004) pioneered the study of deterministic KPS for DSNs
- they suggested using certain combinatorial designs (namely, **generalized quadrangles** and **symmetric BIBDs**) to construct KPS
- a drawback of their approach was a lack of flexibility: they could construct KPS only for fairly constrained parameter situations
- we describe an approach that is much more flexible

Advantages of Deterministic KPS

Deterministic KPS have several potential advantages:

- **Simpler set-up** No random number generator is required for key assignment; simple formulas dictate which keys are given to which nodes.
- **Guaranteed optimal connectivity** We can guarantee that every key is assigned to exactly vk/b nodes, which optimizes connectivity.
- **No need to verify expected properties of the DSN** Randomized KPS have desirable properties with high probability, but there are no guarantees, e.g., due to a “bad” choice of random numbers.
- **Simpler shared-key discovery and path-key establishment** The complexity of these algorithms can be reduced to $O(1)$ (as compared to the $O(k)$ or $O(k \log k)$ algorithms in the randomized case).

Mathematical Model: Combinatorial Set Systems

- a **set system** is a pair (X, \mathcal{A}) , where the elements of X are called **points** and \mathcal{A} is a finite set of subsets of X , called **blocks**
- the **degree** of a point $x \in X$ is the number of blocks containing x
- (X, \mathcal{A}) is **regular** (of degree r) if all points have the same degree, r
- if all blocks have the same size, say k , then (X, \mathcal{A}) is said to be **uniform** (of rank k)
- a **(v, b, r, k) -design** is a set system (X, \mathcal{A}) where $|X| = v$, $|\mathcal{A}| = b$, that is uniform of rank k and regular of degree r
- we associate each block of the set system with a node in the DSN
- the points in the block are the key identifiers of the keys given to the corresponding node

Toy Example

We list the blocks in a $(7, 7, 3, 3)$ -design and the keys in a corresponding KPS:

node	block	key assignment
N_1	{1, 2, 4}	k1, k2, k4
N_2	{2, 3, 5}	k2, k3, k5
N_3	{3, 4, 6}	k3, k4, k6
N_4	{4, 5, 7}	k4, k5, k7
N_5	{1, 5, 6}	k1, k5, k6
N_6	{2, 6, 7}	k2, k6, k7
N_7	{1, 3, 7}	k1, k3, k7

The actual values of keys are secret, but the lists of key identifiers (i.e., the blocks) are not secret.

Configurations

- in order to optimize the connectivity of the DSN, it is necessary and sufficient that there do not exist two nodes containing more than one common key
- this motivates the use of set systems known as **configurations**
- a (v, b, r, k) -configuration is a (v, b, r, k) -design such that any two blocks contain at most one common point
- it is easy to see that $bk = vr$ holds in any (v, b, r, k) -design
- $v \geq r(k - 1) + 1$ and $b \geq k(r - 1) + 1$ are additional necessary conditions for existence of a (v, b, r, k) -configuration

μ -Common Intersection Designs

- the last requirement (# 4) of a KPS for a DSN motivates the following additional property we consider for configurations
- a (v, b, r, k) -configuration is a μ -common intersection design if the following holds for all pairs of blocks A_1 and A_2 with $A_1 \cap A_2 = \emptyset$:

$$|\{A_3 \in \mathcal{A} : A_1 \cap A_3 \neq \emptyset \text{ and } A_2 \cap A_3 \neq \emptyset\}| \geq \mu$$

- for short, we write $(v, b, r, k; \mu)$ -CID
- this suggests the following combinatorial problem:
- given parameters (v, b, r, k) such that at least one (v, b, r, k) -configuration exists, find the **largest** integer μ such that there exists a $(v, b, r, k; \mu)$ -CID

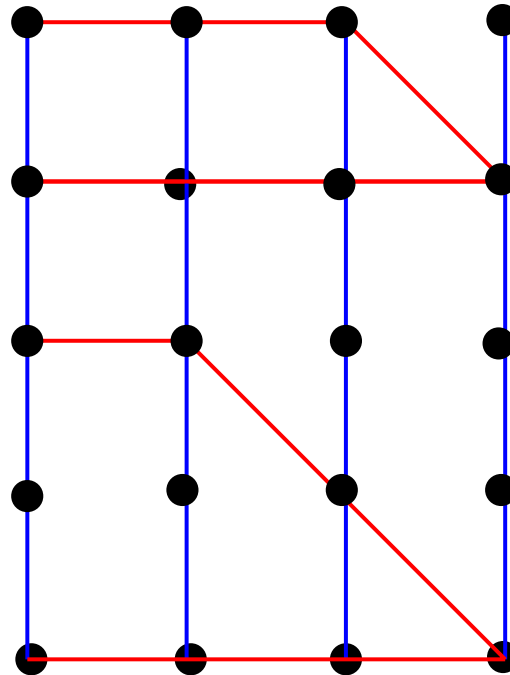
Terminology for Set Systems and KPS

KPS	Set System	Parameter
network size	number of blocks	b
size of key pool	number of points	v
number of keys per node	block-size (rank)	k
number of nodes per key	degree of a point	r
number of two-hop paths connecting two nodes	number of blocks intersecting two disjoint blocks	μ

Transversal Designs

- Let n and k be positive integers such that $2 \leq k \leq n$.
- a **transversal design $TD(k, n)$** is a triple $(X, \mathcal{H}, \mathcal{A})$, where X is a finite set of cardinality kn , \mathcal{H} is a partition of X into k parts (called **groups**) of size n , and \mathcal{A} is a set of k -subsets of X (called **blocks**), which satisfy the following properties:
 1. $|H \cap A| = 1$ for every $H \in \mathcal{H}$ and every $A \in \mathcal{A}$, and
 2. every pair of elements of X from different groups occurs in exactly one block in \mathcal{A} .
- a $TD(k, n)$ is equivalent to a set of $k - 2$ **mutually orthogonal Latin squares** of order n
- a $TD(k, n)$ is a $(kn, n^2, n, k; k^2 - k)$ -CID

Some Blocks in a TD (Diagram)



Groups are represented as vertical blue lines, and blocks are represented as red lines. Each block is a transversal of the groups.

A Construction for Transversal Designs

- suppose that p is prime and $2 \leq k \leq p$

- define

$$X = \{0, \dots, k-1\} \times \mathbb{Z}_p$$

- for every ordered pair $(i, j) \in \mathbb{Z}_p \times \mathbb{Z}_p$, define a block

$$A_{i,j} = \{(x, ix + j \bmod p) : 0 \leq x \leq k-1\}$$

- let

$$\mathcal{A} = \{A_{i,j} : (i, j) \in \mathbb{Z}_p \times \mathbb{Z}_p\}$$

- (X, \mathcal{A}) is a $\text{TD}(k, p)$, which is a $(kp, p^2, p, k; k^2 - k)$ -CID
- the construction can be adapted to any finite field \mathbb{F}_q , where q is a prime power
- KPS constructed from these transversal designs are called **linear schemes**

Example

Suppose we take $p = 5$ and $k = 4$; then we construct a $\text{TD}(4, 5)$:

$$\begin{array}{lll}
 A_{0,0} = \{00,10,20,30\} & A_{0,1} = \{01,11,21,31\} & A_{0,2} = \{02,12,22,32\} \\
 A_{0,3} = \{03,13,23,33\} & A_{0,4} = \{04,14,24,34\} & A_{1,0} = \{00,11,22,33\} \\
 A_{1,1} = \{01,12,23,34\} & A_{1,2} = \{02,13,24,30\} & A_{1,3} = \{03,14,20,31\} \\
 A_{1,4} = \{04,14,24,34\} & A_{2,0} = \{00,12,24,31\} & A_{2,1} = \{01,13,20,32\} \\
 A_{2,2} = \{02,14,21,33\} & A_{2,3} = \{03,10,22,34\} & A_{2,4} = \{04,11,23,30\} \\
 A_{3,0} = \{00,13,21,34\} & A_{3,1} = \{01,14,22,30\} & A_{3,2} = \{02,10,23,31\} \\
 A_{3,3} = \{03,11,24,32\} & A_{3,4} = \{04,12,20,33\} & A_{4,0} = \{00,14,23,32\} \\
 A_{4,1} = \{01,10,24,33\} & A_{4,2} = \{02,11,20,34\} & A_{4,3} = \{03,12,21,30\} \\
 A_{4,4} = \{04,13,22,31\} & &
 \end{array}$$

Two-hop Paths

- suppose we use a $(v, b, r, k; \mu)$ -CID for key predistribution in a DSN
- assume that the sensor nodes are distributed in the Euclidean plane in a random way and the range covered by each node forms a circle of fixed radius whose center is that node (we call this circle a **neighbourhood** of the given sensor node)
- suppose that N_i and N_j are two nodes that are in each other's neighbourhood
- the probability that N_i and N_j share a common key is

$$p_1 = \frac{k(r-1)}{b-1}$$

Two-hop Paths (cont.)

- let η denote the number of nodes in the intersection of the neighbourhoods of the two nodes N_i and N_j
- the probability (denoted by p_2) that N_i and N_j do not share a common key, but there exists a node N_h in the intersection of their neighbourhoods such that N_h shares a key with both N_i and N_j , is estimated as follows:

$$p_2 \approx \left(1 - \frac{k(r-1)}{b-1}\right) \times \left(1 - \left(1 - \frac{\mu}{b-2}\right)^\eta\right)$$

- the probability that N_i is connected to N_j via a path of length one or two is roughly $p_1 + p_2$

Example

- suppose we use a TD(30, 49) as a key predistribution scheme
- this transversal design is a (1470, 2401, 49, 30; 870)-CID
- we can support 2401 nodes in the resulting DSN, and every node is required to store 30 keys
- suppose that nodes are distributed in a physical region in such a way that $\eta \geq 20$
- then we have

$$p_1 = 0.6,$$

$$p_2 \approx 0.39995, \quad \text{and}$$

$$p_1 + p_2 \approx 0.99995$$

- hence, in the resulting DSN, the probability that two nearby nodes are not connected in one or two hops is less than 0.00005

Example (cont.)

Even for smaller values of η , we achieve good local connectivity:

η	p_1	p_2	$p_1 + p_2$
1	0.6	0.145	0.745
2	0.6	0.237	0.837
3	0.6	0.296	0.896
4	0.6	0.334	0.934
5	0.6	0.358	0.958
10	0.6	0.396	0.996
15	0.6	0.3995	0.9995
20	0.6	0.39995	0.99995

Resiliency

- suppose that N_h , N_i and N_j all have a common key L and node N_h is compromised
- assuming that the key predistribution is done using a (v, b, r, k) -configuration, we conclude that N_i and N_j can no longer communicate (directly) in a secure manner
- therefore, we say that the compromise of N_h **affects** the link from N_i to N_j
- an arbitrary link between two nodes is affected with probability $(r - 2)/(b - 2)$ by the compromise of some other random node
- the compromise of s random nodes will affect a given link with probability roughly equal to

$$fail(s) = 1 - \left(1 - \frac{r - 2}{b - 2}\right)^s$$

Example

- as before, suppose we construct a linear KPS using a $(1470, 2401, 49, 30; 870)$ -CID, so we have $b = 2401$ and $r = 49$
- $fail(10) \approx 0.1795$, so any given link is affected with a probability of about 18% when ten random nodes are compromised
- for smaller values of s , we achieve better resiliency:

s	$fail(s)$
1	0.0196
2	0.0388
3	0.0576
4	0.0761
6	0.1119
8	0.1464

Shared-key Discovery for Linear Schemes

- a randomized KPS has no “structure”
- hence, shared-key discovery between two nodes N_i and N_j typically requires the nodes to exchange the list of indices of the keys they hold in order for them to be able to determine if they share a common key
- this increases the communication complexity of the protocol, decreases battery life, etc.
- an advantage of using deterministic KPS is that they may have a compact and efficient algebraic description
- this may yield nice algorithms for shared-key discovery, in which very little information needs to be broadcasted
- suppose we use a linear KPS based on a transversal design $TD(k, p)$
- in the resulting DSN, each node is identified by an ordered pair $(i, j) \in \mathbb{Z}_p \times \mathbb{Z}_p$

Shared-key Discovery and Path-key Establishment

- it is sufficient for two nodes to exchange their identifiers
- two nodes, say $N_{(i,j)}$ and $N_{(i',j')}$, can independently determine if they share a common key in $O(1)$ time, as follows:
 1. If $i = i'$ (and hence $j \neq j'$) then $N_{(i,j)}$ and $N_{(i',j')}$ do not share a common key
 2. Otherwise, compute $x = (j' - j)(i - i')^{-1} \bmod p$. If $0 \leq x \leq k - 1$, then $N_{(i,j)}$ and $N_{(i',j')}$ share the common key $k(x, ix+j)$. If $x \geq k$, then $N_{(i,j)}$ and $N_{(i',j')}$ do not share a common key.
- path-key establishment: if the two nodes $N_{(i,j)}$ and $N_{(i',j')}$ do not share a common key, then they can easily determine if there are two-hop paths joining them, given the identifiers of all the nodes in the intersection of their neighborhoods

Quadratic Schemes

- the only (?) drawback of linear schemes is that they might not be able to support networks of sufficiently large size, especially when p_1 is “large” and k is “small” (these conditions force n to be small)
- therefore we also proposed a class of **quadratic schemes**
- suppose that p is prime and $3 \leq k \leq p$, and define

$$X = \{0, \dots, k-1\} \times \mathbb{Z}_p$$

- for every ordered triple $(i_1, i_2, i_3) \in \mathbb{Z}_p^3$, define a block

$$A_{i_1, i_2, i_3} = \{(x, i_1 x^2 + i_2 x + i_3 \bmod p) : 0 \leq x \leq k-1\}$$

and let

$$\mathcal{A} = \{A_{i_1, i_2, i_3} : (i_1, i_2, i_3) \in \mathbb{Z}_p^3\}$$

- two nodes share a key iff the corresponding blocks have **two** common points

References

Most of the work in this talk is from the following papers:

- **J. Lee and D. R. Stinson.** Common intersection designs. *Journal of Combinatorial Designs* 14 (2006), 251-269.
- **J. Lee and D. R. Stinson.** A combinatorial approach to key predistribution for distributed sensor networks. *IEEE Wireless Communications and Networking Conference*, CD-ROM, 2005, paper PHY53-06, 6 pp. [Invited paper.]
- **J. Lee and D. R. Stinson.** On the construction of practical key predistribution schemes for distributed sensor networks using combinatorial designs. Submitted.

These papers are available from:

<http://www.cacr.math.uwaterloo.ca/~dstinson/>