Research in Information Security (CSE540)
Assignment 2

Total Marks: 100

Hard deadline: October 15, 2016 (Saturday), 11:25 PM

Implementation of the Blom's single-space key pre-distribution scheme [R. Blom. "An optimal class of symmetric key generation systems," in Advances in Cryptology: Proceedings of EUROCRYPT'84, Lecture Notes in Computer Science, Springer-Verlag, volume 209, pages 235-238, 1985].

Outline of the assignment

1. Consider a deployment or target field of area $500\,m \times 500\,m$. In this area, a total of n sensor nodes $($for example, $n = 600)$ are randomly deployed in the target field.

2. After deployment of nodes, each node will compute its neighbor sensor nodes within its communication range, say $25\,m$. Prepare a neighbor list of each sensor node i. Let this list be NL_i.

3. Plot the positions of all deployed sensor nodes in a graph and stored in a pdf file, say deployment_rollno.pdf.

4. Now, apply the Blom's key distribution scheme on this network. Generate a primitive root g of the finite field $GF(q)$, where q is a large prime. After that take the matrix G as

$$G = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ g & g^2 & g^3 & \cdots & g^n \\ g^2 & (g^2)^2 & (g^3)^2 & \cdots & (g^n)^2 \\ & & & \vdots & \\ g^\lambda & (g^2)^\lambda & (g^3)^\lambda & \cdots & (g^n)^\lambda \end{pmatrix}.$$

   Display this matrix.

5. Finally, output all pairwise keys between neighbor nodes in the network in a file, say keys.txt as follows:
   Node i: $NL_i = \{j_1, j_2, \ldots, j_d\}$
   $Key(i, j_1) = \ldots, \ldots Key(i, j_d) = \ldots$.

Instructions:

Submission guideline Upload the assignment in soft copy of the C code file (assign2_rollno.c), figure of deployment of sensor nodes in the network (deployment_rollno.pdf), keys.txt and a README file in a ZIP file (assign2_rollno.zip) in the course portal (moodle).
In the C file, please write your name, roll number and assignment 2 as comments.

APPENDIX: The single-space key pre-distribution scheme

Blom proposed a key pre-distribution scheme that allows any pair of sensor nodes in a sensor network to be able to establish a pairwise secret key. This scheme is described as follows.

In the *key pre-distribution phase*, the (key) setup server first constructs a $(\lambda + 1) \times n$ matrix G over a finite field $F_q = GF(q)$, where q is prime and n is the number of sensor nodes in the network. Here G is considered as public information, that is, any sensor node can know the contents of G and even adversaries are allowed to know G. Then the setup server generates a random $(\lambda + 1) \times (\lambda + 1)$ symmetric matrix D over $GF(q)$ and computes another matrix A as $A = (D \cdot G)^T$, where $(D \cdot G)^T$ is the transpose of the matrix $D \cdot G$. We note that A is of order $n \times (\lambda + 1)$. Matrix D is kept secret and thus it is not disclosed to any sensor node also. We see that $A \cdot G = (D \cdot G)^T \cdot G = G^T \cdot D^T \cdot G = G^T \cdot D \cdot G = (A \cdot G)^T$, since D is a symmetric matrix. Thus, $A \cdot G$ is also a symmetric matrix of order $n \times n$. If we let $K = A \cdot G$ and $K = (k_{ij})_{n \times n}$, then we have $k_{ij} = k_{ji}$. Finally, the setup server loads the following informations to each deployed sensor node i ($i = 1, 2, \ldots, n$):

(i) the i-th row of matrix A, and

(ii) the i-th column of matrix G.

After deployment, every sensor node locates its physical neighbors within its communication range. When two neighbor sensor nodes i and j want to establish a pairwise secret key between them, they need to exchange their columns of G and then they compute the secret keys k_{ij} and k_{ji} respectively, using their private rows of A. Since the matrix K is symmetric, so $k_{ij} = k_{ji}$. Again, since G is public, the nodes can transmit their columns in plaintext only. Nodes i and j store this computed key k_{ij} $(= k_{ji})$ for their future secret communications.

It has been proved that this scheme is λ-secure if any $\lambda + 1$ columns of G are linearly independent. This λ-secure property guarantees that no nodes other than i and j can compute their secret key k_{ij} or k_{ji}, if no more than λ nodes are compromised. We note that the security of this scheme is also similar to that for the polynomial-based key pre-distribution scheme described in Section 2.8.

Since each pairwise key is represented by an element in the finite field $GF(q)$, if the length of pairwise keys is 64-bits, then we have to choose q as the smallest prime number that is larger than 2^{64}. Let g be a primitive element of $GF(q)$ and $n < q$. From the property of the primitive elements, we note that $g^i \neq g^j$ if $i \neq j$. A feasible G can be designed as follows:

$$G = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ g & g^2 & g^3 & \cdots & g^n \\ g^2 & (g^2)^2 & (g^3)^2 & \cdots & (g^n)^2 \\ & & & \vdots & \\ g^\lambda & (g^2)^\lambda & (g^3)^\lambda & \ldots & (g^n)^\lambda \end{pmatrix}.$$

Since G is a Vandermonde matrix, it can be easily shown that any $\lambda + 1$ columns of G are linearly independent when $g, g^2, g^3, \ldots, g^n$ are all distinct. In practice, G can be generated by the primitive element g of $GF(q)$. Although the value of λ can improve the security property of this scheme, it is not feasible due to the limited memory in sensors.