Research Journal of Applied Sciences, Engineering and Technology 11(7): 780-787, 2015
DOI: 10.19026/rjaset.11.2041
ISSN: 2040-7459; e-ISSN: 2040-7467
© 2015 Maxwell Scientific Publication Corp.
Submitted: April 23, 2015 Accepted: May 22, 2015 Published: November 05, 2015

Research Article

Application of Modulo Key-Predistribution Protocol

[1]K. Rajadurai, [2]T. Kavitha and [3]V.J. Subashini
[1]Optum Golbal Solutions, United Health Group, Bangalore,
[2]New Prince Shri Bhavani College of Engineering and Technology, Anna University, Chennai,
[3]Jerusalem College of Engineering, Anna University, Chennai, India

Abstract: The aim of the study is to analyze the application of the Modulo Key Pre-Distribution (MKPD) protocol which is developed for Wireless sensor network (WSN). WSN is a collection of a large number of sensor nodes with limited resources. Bootstrapping secure communication among sensor nodes deployed in hostile environment is an important and challenging problem. An adversary may physically capture some sensors to compromise their stored sensitive data and secret keys which are used to attain confidentiality of the sensitive data. A common approach to solve problem is to use a key pre-distribution scheme in which each sensor node is assigned a subset of keys (key chain), selected from key pool prior to deployment. The keys are carefully selected that have high key sharing probability and resistance against node capture. We use deterministic key distribution scheme based on modulo function to establish the pair-wise keys among the sensor nodes. This study focuses on the application of this Modulo Key Pre-Distribution (MKPD) protocol to a group communication. It provides good key connectivity between the sensor nodes belongs to that group. And also opportunity available to increase the scalability in MKPD protocol by complementary design is also explained.

Keywords: Complementary design, group communication, modulo function, modulo key pre-distribution, security, trade, wireless sensor networks

INTRODUCTION

A Wireless Sensor Network (WSN) is a collection of spatially distributed autonomous sensors to cooperatively monitor physical or environmental condition. Each sensor node is a small, inexpensive wireless device with limited battery power, memory storage, data processing capacity and short radio transmission range (Akyildiz *et al*., 2002; Pottie and Kaiser, 2000). The sensor networks are used in many applications, like military and civilian services like battlefield surveillance, flood detection and various home applications; are often deployed in hostile environments. Establishing secure communication among sensor nodes deployed in hostile environment is an important and challenging problem (Lee *et al*., 2007). To secure the communication among the nodes, a general approach is to use a cryptographic algorithm. When considering the constraints of the node and network, asymmetric cryptography is more expensive compared to symmetric key cryptography. So, symmetric key cryptography is preferred. Strength of the any cryptographic algorithm relies on the efficient key management techniques. The main task of key management technique is the safe distribution of secret key to communicating nodes before deployment (key pre distribution) or safe agreement of keys between communicating nodes after deployment.

Environments, where sensor nodes are deployed, can be controlled or uncontrolled such as hostile or disaster areas, toxic regions. If the environment is known under control, deployment may be achieved manually to establish an infrastructure. However, manual deployments become infeasible or even impossible as the number of the nodes increases. If the environment is uncontrolled or the WSN is very large, deployment has to be performed by randomly scattering the sensor nodes to target area. It may be possible to provide denser sensor deployment at certain spots, but exact positions of the sensor nodes cannot be controlled. Thus, network topology cannot be known precisely prior to the deployment. Thus the key distribution technique loads keys to sensor nodes before deployment i.e., key pre distribution is employed in sensor network.

One way of pre-distribution is to load all the nodes with a single master key. This results in an optimal storage. However if one node is compromised, then the entire network becomes insecure. At the other extreme, each pair of nodes can share a unique key called pair-wise key which increases resiliency at the same time memory requirement also increases. In order to overcome this problem, key pre-distribution is employed. It consists of three phases:

Corresponding Author: K. Rajadurai, Optum Golbal Solutions, United Health Group, Bangalore, India

Key pre-distribution: In this offline step, subset of key are preloaded in to the sensor nodes from the key pool prior to deployment. Subset selection from the key pool can be random or deterministic method.

Shared key discovery process: After deployment, the sensors need to find if they share a common key with their neighbours.

Path key establishment: If shared key discovery process fails, node has to discover a path using intermediate nodes between the two terminal nodes. The detailed classification of key distribution schemes were discussed by Kavitha and Sridharan (2013a).

Here we use a deterministic key pre-distribution scheme using modulo function for the distributed homogeneous network group communication. And we prove that modulo trades provides good connectivity and resilience among the sensor nodes belongs to that group compared to the combinatorial trades. And also this study explains the opportunity available to increase the scalability in MKPD protocol by complementary design.

LITERATURE REVIEW

First key pre-distribution scheme using probabilistic model is proposed in Eschenauer and Gligor (2002) which is enhanced by having q common keys to establish a link key (Pietro *et al*., 2003); it increases the resiliency. With probabilistic model, deployment knowledge is combined by grid based deployment method (Mehta *et al*., 2005) which increases the local connectivity. Pseudo random function is utilized by Kausar *et al*. (2008) which eliminate the communication overhead that reduces the energy consumption of node. Node ID using hash chain based key distribution algorithm is introduced in Mehta *et al*. (2005) and Kausar *et al*. (2008) in contrast to Po-Jen *et al*. (2005 scheme where the group ID is used. Ren *et al*. (2006) and Shan and Liu (2008) combines probabilistic approach with hash chain technique which increases the scalability. Key mapping technique using expected resident point of a node according to PDF is proposed by Zeen *et al*. (2007). Reusable key pool based probabilistic key distribution scheme is proposed by Levi *et al*. (2010) which increases the scalability. Bechkit *et al*. (2010) proposed a scheme which uses the tree based probabilistic key distribution scheme and hash function; it increases the resiliency. Trade property of combinatorial design is used in Ruj *et al*. (2011), provides the pair-wise key for a pair of node uniquely, which decreases the connectivity and increases the resiliency.

METHOD-MODULO KEY PRE-DISTRIBUTION PROTOCOL

Kavitha and Sridharan (2011) first proposed a Modulo Key Pre-distribution Protocol for WSN and their connectivity, resilience, sclability are analyzed using analytical results. But the problem is when the PRP set size increases, the number of terms also increases linearly and the expression for each term also increases. So, the expression for connectivity and resilience grows linearly with PRP set size dynamically which increases the complexity in implementing such a dynamic expression. So in order to reduce the expression, probabilistic method is used to derive the expression for connectivity and resilience by Kavitha and Sridharan (2013b).

The concept behind this modulo scheme is explained as follows. The modulo set m_i of (n+1) elements are selected where i= 0, 1, 2,…., n. such that all are relatively prime numbers, i.e.:

$$gcd(m_i, m_j) = 1\, for\ i \neq j. \tag{1}$$

The elements of m_i (Z_{mi}) is formed from the key pool $P_{i.}$ The elements can be given as:

$$Z_{mi} = \{0, 1, \ldots\ldots (m_i - 1)\} \tag{2}$$

In that, there is a unique integer id (William, 2013) where $id \in Z_M$ and Z_M = {0,1,…..(M-1)}, here M= $\prod_{i=0}^{n} m_i$. For any integer id in Z_M represented by a unique (n+1) tuble whose elements are in Z_{mi} using the correspondence:

$$id \leftrightarrow (a_0, a_1 \ldots a_i \ldots a_n)\ where\ a_i \in Z_{mi} and\ a_i = id\ mod\ m_i\, for\ 0 \leq i \leq n. \tag{3}$$

Example 1: Let the PRP numbers be (m_0, m_1, m_2) = (2, 3, 5) and the residues be (r_0, r_1, r_2) = (1, 2, 4). Then the congruence equations become (x ≡ 1 mod 2), (x ≡ 2 mod 3) and (x ≡ 4 mod 5). The value of x = 29 can be found by CRT as prescribed by the above formulas. Here, M = 30, so Z_M = {0, 1, 2, 3 ……….29}. Let an integer $A \in Z_M = 16$ can be expressed as a unique 3-tuple as (a_0, a_1, a_2) ↔ (0, 1, 1).

Working of this basic modulo scheme is given below. The 'n+1' numbers of Pair-wise Relatively Prime numbers (PRP) modulo m_i, are selected where M = m_0 m_1….m_n. Then the 'n+1' key pools are arranged by using the Eq. (4):

$$P_i \rightarrow (i,0)(i,1)(i,2)\ldots.(i,m_i - 1) \tag{4}$$

Using the key identifier id, the Key Chain (KC) is generated where 0 ≤ id < M – 1.

$$KC_{id} = [(0, a_0)(1, a_1)\ldots.(n-1, a_{n-1})] \tag{5}$$

where, a_i = id mod m_i for 0 ≤ i ≤ n. The last step is to assign the key chain with key identifier to sensor node specified by the id.

Table 1: Key Chain set of size 105 of (3, 5, 7)

id	KC_{id}	$(0,a_0)$	$(1,a_1)$	$(2,a_2)$	*id*	KC_{id}	$(0,a_0)$	$(1,a_1)$	$(2,a_2)$
0	KC_0	(0,0)	(1,0)	(2,0)	1	KC_1	(0,1)	(1,1)	(2,1)
2	KC_2	(0,2)	(1,2)	(2,2)	3	KC_3	(0,0)	(1,3)	(2,3)
4	KC_4	(0,1)	(1,4)	(2,4)	5	KC_5	(0,2)	(1,0)	(2,5)
6	KC_6	(0,0)	(1,1)	(2,6)	7	KC_7	(0,1)	(1,2)	(2,0)
8	KC_8	(0,2)	(1,3)	(2,1)	9	KC_9	(0,0)	(1,4)	(2,2)
10	KC_{10}	(0,1)	(1,0)	(2,3)	11	KC_{11}	(0,2)	(1,1)	(2,4)
12	KC_{12}	(0,0)	(1,2)	(2,5)	13	KC_{13}	(0,1)	(1,3)	(2,6)
14	KC_{14}	(0,2)	(1,4)	(2,0)	15	KC_{15}	(0,0)	(1,0)	(2,1)
16	KC_{16}	(0,1)	(1,1)	(2,2)	17	KC_{17}	(0,2)	(1,2)	(2,3)
18	KC_{18}	(0,0)	(1,3)	(2,4)	19	KC_{19}	(0,1)	(1,4)	(2,5)
20	KC_{20}	(0,2)	(1,0)	(2,6)	21	KC_{21}	(0,0)	(1,1)	(2,0)
22	KC_{22}	(0,1)	(1,2)	(2,1)	23	KC_{23}	(0,2)	(1,3)	(2,2)
24	KC_{24}	(0,0)	(1,4)	(2,3)	25	KC_{25}	(0,1)	(1,0)	(2,4)
26	KC_{26}	(0,2)	(1,1)	(2,5)	27	KC_{27}	(0,0)	(1,2)	(2,6)
28	KC_{28}	(0,1)	(1,3)	(2,0)	29	KC_{29}	(0,2)	(1,4)	(2,1)
30	KC_{30}	(0,0)	(1,0)	(2,2)	31	KC_{31}	(0,1)	(1,1)	(2,3)
32	KC_{32}	(0,2)	(1,2)	(2,4)	33	KC_{33}	(0,0)	(1,3)	(2,5)
34	KC_{34}	(0,1)	(1,4)	(2,6)	35	KC_{35}	(0,2)	(1,0)	(2,0)
36	KC_{36}	(0,0)	(1,1)	(2,1)	37	KC_{37}	(0,1)	(1,2)	(2,2)
38	KC_{38}	(0,2)	(1,3)	(2,3)	39	KC_{39}	(0,0)	(1,4)	(2,4)
40	KC_{40}	(0,1)	(1,0)	(2,5)	41	KC_{41}	(0,2)	(1,1)	(2,6)
42	KC_{42}	(0,0)	(1,2)	(2,0)	43	KC_{43}	(0,1)	(1,3)	(2,1)
44	KC_{44}	(0,2)	(1,4)	(2,2)	45	KC_{45}	(0,0)	(1,0)	(2,3)
46	KC_{46}	(0,1)	(1,1)	(2,4)	47	KC_{47}	(0,2)	(1,2)	(2,5)
48	KC_{48}	(0,0)	(1,3)	(2,6)	49	KC_{49}	(0,1)	(1,4)	(2,0)
50	KC_{50}	(0,2)	(1,0)	(2,1)	51	KC_{51}	(0,0)	(1,1)	(2,2)
52	KC_{52}	(0,1)	(1,2)	(2,3)	53	KC_{53}	(0,2)	(1,3)	(2,4)
54	KC_{54}	(0,0)	(1,4)	(2,5)	55	KC_{55}	(0,1)	(1,0)	(2,6)
56	KC_{56}	(0,2)	(1,1)	(2,0)	57	KC_{57}	(0,0)	(1,2)	(2,1)
58	KC_{58}	(0,1)	(1,3)	(2,2)	59	KC_{59}	(0,2)	(1,4)	(2,3)
60	KC_{60}	(0,0)	(1,0)	(2,4)	61	KC_{61}	(0,1)	(1,1)	(2,5)
62	KC_{62}	(0,2)	(1,2)	(2,6)	63	KC_{63}	(0,0)	(1,3)	(2,0)
64	KC_{64}	(0,1)	(1,4)	(2,1)	65	KC_{65}	(0,2)	(1,0)	(2,2)
66	KC_{66}	(0,0)	(1,1)	(2,3)	67	KC_{67}	(0,1)	(1,2)	(2,4)
68	KC_{68}	(0,2)	(1,3)	(2,5)	69	KC_{69}	(0,0)	(1,4)	(2,6)
70	KC_{70}	(0,1)	(1,0)	(2,0)	71	KC_{71}	(0,2)	(1,1)	(2,1)
72	KC_{72}	(0,0)	(1,2)	(2,2)	73	KC_{73}	(0,1)	(1,3)	(2,3)
74	KC_{74}	(0,2)	(1,4)	(2,4)	75	KC_{75}	(0,0)	(1,0)	(2,5)
76	KC_{76}	(0,1)	(1,1)	(2,6)	77	KC_{77}	(0,2)	(1,2)	(2,0)
78	KC_{78}	(0,0)	(1,3)	(2,1)	79	KC_{79}	(0,1)	(1,4)	(2,2)
80	KC_{80}	(0,2)	(1,0)	(2,3)	81	KC_{81}	(0,0)	(1,1)	(2,4)
82	KC_{82}	(0,1)	(1,2)	(2,5)	83	KC_{83}	(0,2)	(1,3)	(2,6)
84	KC_{84}	(0,0)	(1,4)	(2,0)	85	KC_{85}	(0,1)	(1,0)	(2,1)
86	KC_{86}	(0,2)	(1,1)	(2,2)	87	KC_{87}	(0,0)	(1,2)	(2,3)
88	KC_{88}	(0,1)	(1,3)	(2,4)	89	KC_{89}	(0,2)	(1,4)	(2,5)
90	KC_{90}	(0,0)	(1,0)	(2,6)	91	KC_{91}	(0,1)	(1,1)	(2,0)
92	KC_{92}	(0,2)	(1,2)	(2,1)	93	KC_{93}	(0,0)	(1,3)	(2,2)
94	KC_{94}	(0,1)	(1,4)	(2,3)	95	KC_{95}	(0,2)	(1,0)	(2,4)
96	KC_{96}	(0,0)	(1,1)	(2,5)	97	KC_{97}	(0,1)	(1,2)	(2,6)
98	KC_{98}	(0,2)	(1,3)	(2,0)	99	KC_{99}	(0,0)	(1,4)	(2,1)
100	KC_{100}	(0,1)	(1,0)	(2,2)	101	KC_{101}	(0,2)	(1,1)	(2,3)
102	KC_{102}	(0,0)	(1,2)	(2,4)	103	KC_{103}	(0,1)	(1,3)	(2,5)
104	KC_{104}	(0,2)	(1,4)	(2,6)					

Example 2:

Parameter selection: Consider the network size N as 100. Let us take 'three' numbers of PRPs as $m_0 = 3$, $m_1 = 5$, $m_2 = 7$ and M becomes 105 such that the step-1b of MKPD protocol (Kavitha and Sridharan, 2013b) i.e., $N \leq M$ and $M = \prod_{i=0}^{n} m_i$ is satisfied.

Key pool arrangement: Here the number of PRPs taken is three. So there are three key pools, which are arranged as follows:

P_0 → (0, 0) (0, 1) (0, 2)
P_1 → (1, 0) (1, 1) (1, 2) (1, 3) (1, 4)
P_2 → (2, 0) (2, 1) (2, 2) (2, 3) (2.4) (2, 5) (2, 6)

KC generation: Generate the key chain by using the Eq. (5). Here, *id* varies from 0 to 104. So, totally 105 key chains can be generated. Since PRP numbers are three, key chain size is also three. The set of key chains $(Z_{KC_{id}})$ are given in Table 1.

Then, the KC_{id} is assigned to sensor node *id*: Kavitha and Sridharan (2013a) calculate the probability of sharing at-least one key for any pair of key chain and fraction of links compromised when N_c number of nodes compromised. They have shown that basic modulo scheme supports large network with small key pool and connectivity increases with network size but decreases the resiliency and memory requirement increases linearly with logarithmic increase in network.

Table 2: Group key distribution

Trade length q	Number of ways the key pool set is chosen for trade of length q	Keypool chosen for trade t_q	Group size $\|G\|$	Number groups of G_n
1	4	(P_0)	$m_1m_2m_3$	m_0
		(P_1)	$m_0m_2m_3$	m_1
		(P_2)	$m_0m_1m_3$	m_2
		(P_3)	$m_0m_1m_2$	m_3
2	6	$(P_{0,} P_1)$	m_2m_3	m_0m_1
		$(P_{0,} P_2)$	m_1m_3	m_0m_2
		$(P_{0,} P_3)$	m_1m_2	m_0m_3
		$(P_{1,} P_2)$	m_0m_3	m_1m_2
		$(P_{1,} P_3)$	m_0m_2	m_1m_3
		$(P_{2,} P_3)$	m_0m_1	m_2m_3
3	4	$(P_{0,} P_{1,} P_2)$	m_3	$m_0m_1m_2$
		$(P_{0,} P_{2,} P_3)$	m_1	$m_0m_2m_3$
		$(P_{0,} P_{1,} P_3)$	m_2	$m_0m_1m_3$
		$(P_{1,} P_{2,} P_3)$	m_0	$m_1m_2m_3$

RESULTS AND DISCUSSION

Application of MKPD protocol to group communication: A KPD scheme can be mapped to a mathematical set system as explained below. Let (*X, A*) be the set system with a set of *X* elements mapped to a pool of key identifiers and the set of subsets *A* formed from the elements of *X*. Each subset belonging to *A* is called a block, which can be mapped to a key chain of a sensor node. In a BIBD, also known as 2-design, the parameters of the design are *v* (the number of elements of *X*), *b* (the number of blocks), *r* (the number of blocks containing a given element), *k* (the number of elements in a block) and λ (the number of blocks containing a given pair of elements). For a combinatorial *t*-design, every *t*-subset (trade) of *X* occurs in λ blocks. The combinatorial designs with *t* = 2 and *t* = 3 can be used to establish unique pair-wise keys and triple keys respectively between the sensor nodes. The scenario where three nodes want to communicate securely is called as the triple key distribution (Ruj *et al*., 2011). It is a special type of group key distribution where the group size is three.

Modulo trade: Similarly, modulo trade is the trade in MKPDP applied to the group key distribution by using the trade. Trade *t* is the subset of the key pools, where each element is taken from different key pools. The trade of length *q* is t_q, having *q* number of elements, where $1 \leq q \leq n$. The number of ways the trade t_q chosen from the key pool set is $(n+1)\text{C}_q$. The number of groups G_n for trade t_q is the product of the size of the key pools from which each j^{th} element of the trade $t_q$is taken.

$$G_n = \prod_{i=1}^{n} m_i, \forall m_i \in t_q. \qquad (6)$$

The number of key chains having t_q is derived from the set of key pools $Z_P \notin t_q$.The number of key chains having t_q, decides the group size $|G|$.

$$|G| = M/G_n \qquad (7)$$

In order to generate the set of key chains having t_q, consider a reference key chain identifier KC_r which has t_q. With respect to the reference key chain identifier KC_r, the set of key chain identifiers KC_{id} having t_q can be given as follows:

$$Z_{KC_{t_q}}\{\cdots KC_{r-2G_n}, KC_{r-G_n}, KC_r, KC_{r+G_n}, KC_{r+2G_n} \cdots\} \qquad (8)$$

where, $0 \leq id \leq M-1$.

For the inter-group communication, $t_q \cap KC_{id}$ keys will be used. The number of keys available for the inter-group communication depends on the length of the trade chosen.

Let PRP set of size four be taken in general as $\{m_0, m_1, m_2, m_3\}$. The corresponding key pools are $\{P_0, P_1, P_2, P_3\}$, respectively. Then, for a possible trade length *q*, the number of ways the key pool sets chosen for the trade of length *q* and its corresponding key pool chosen with its associated number of groups G_n of group size $|G|$, are given in Table 2.

Figure 1 depicts the group size for each possible key pool chosen for the PRP sets of size four. It is observed that when the trade length *q* increases, the size of groups decreases and the number of groups increases.

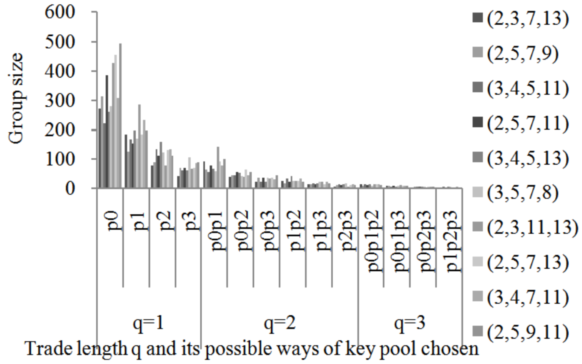

Fig. 1: Combination of key pools for the PRP sets of size four *vs* group size

Group key distribution scheme: The algorithm for the applicability of MKPDP to the group communication is discussed so far and it is given in Table 3:

Example 3:
Inputs: Parameter selection
Consider the parameters chosen as specified in example 1. The PRP numbers be $m_0 = 3$, $m_1 = 5$, $m_2 = 7$ and M = 105.

- Key chain generated in example 1 is shown in Table 1.
- The number of groups G_n and their corresponding group size as given in Table 4
- Divide the key chain given in Table 1 into groups as given in Table 5.

Let the trade length be 2, the group size be 7 and the number of groups be 15. Since the trade length is

Table 3: Group key distribution scheme

1. Generate the set of key chains $Z_{KC_{id}}$ of size n+1 using the algorithm given in (Kavitha and Sridharan 2011)
2. Calculate the number of groups G_n and their corresponding group size as given in Table 2.
3. Select the required number of groups G_n and their corresponding group size.
4. Divide the key chain set $Z_{KC_{id}}$ into G_n groups selected in step3, of size $|G|$, based on the key identifiers of the set of key pools, used for the trade t_q.

Table 4: Group key distribution scheme

Trade length q	Number of ways the key pool set is chosen for trade of length q	Key pool chosen for trade t_q	Group size $\|G\|$	Number of groups G_n
1	3	(P_0)	$m_1m_2 = 35$	$m_0 = 3$
		(P_1)	$m_0m_2= 21$	$m_1= 5$
		(P_2)	$m_0m_1= 15$	$m_2= 7$
2	3	$(P_{0,} P_1)$	$m_2= 7$	$m_0m_1= 15$
		$(P_{0,} P_2)$	$m_1= 5$	$m_0m_2= 21$
		$(P_{1,} P_2)$	$m_0= 3$	$m_1m_2= 35$

Table 5: Group key distribution scheme's key chain

G_n	*id*	KC_{id}	(0,a0)	(1,a1)	(2,a2)	G_n	*id*	KC_{id}	(0,a0)	(1,a1)	(2,a2)
G_1	0	KC_0	(0,0)	(1,0)	(2,0)	G_2	1	KC_1	(0,1)	(1,1)	(2,1)
	15	KC_{15}	(0,0)	(1,0)	(2,1)		16	KC_{16}	(0,1)	(1,1)	(2,2)
	30	KC_{30}	(0,0)	(1,0)	(2,2)		31	KC_{31}	(0,1)	(1,1)	(2,3)
	45	KC_{45}	(0,0)	(1,0)	(2,3)		46	KC_{46}	(0,1)	(1,1)	(2,4)
	60	KC_{60}	(0,0)	(1,0)	(2,4)		61	KC_{61}	(0,1)	(1,1)	(2,5)
	75	KC_{75}	(0,0)	(1,0)	(2,5)		76	KC_{76}	(0,1)	(1,1)	(2,6)
	90	KC_{90}	(0,0)	(1,0)	(2,6)		91	KC_{91}	(0,1)	(1,1)	(2,0)
G_3	2	KC_2	(0,2)	(1,2)	(2,2)	G_4	3	KC_3	(0,0)	(1,3)	(2,3)
	17	KC_{17}	(0,2)	(1,2)	(2,3)		18	KC_{18}	(0,0)	(1,3)	(2,4)
	32	KC_{32}	(0,2)	(1,2)	(2,4)		33	KC_{33}	(0,0)	(1,3)	(2,5)
	47	KC_{47}	(0,2)	(1,2)	(2,5)		48	KC_{48}	(0,0)	(1,3)	(2,6)
	62	KC_{62}	(0,2)	(1,2)	(2,6)		63	KC_{63}	(0,0)	(1,3)	(2,0)
	77	KC_{77}	(0,2)	(1,2)	(2,0)		78	KC_{78}	(0,0)	(1,3)	(2,1)
	92	KC_{92}	(0,2)	(1,2)	(2,1)		93	KC_{93}	(0,0)	(1,3)	(2,2)
G_5	4	KC_4	(0,1)	(1,4)	(2,4)	G_6	5	KC_5	(0,2)	(1,0)	(2,5)
	19	KC_{19}	(0,1)	(1,4)	(2,5)		20	KC_{20}	(0,2)	(1,0)	(2,6)
	34	KC_{34}	(0,1)	(1,4)	(2,6)		35	KC_{35}	(0,2)	(1,0)	(2,0)
	49	KC_{49}	(0,1)	(1,4)	(2,0)		50	KC_{50}	(0,2)	(1,0)	(2,1)
	64	KC_{64}	(0,1)	(1,4)	(2,1)		65	KC_{65}	(0,2)	(1,0)	(2,2)
	79	KC_{79}	(0,1)	(1,4)	(2,2)		80	KC_{80}	(0,2)	(1,0)	(2,3)
	94	KC_{94}	(0,1)	(1,4)	(2,3)		95	KC_{95}	(0,2)	(1,0)	(2,4)
G_7	6	KC_6	(0,0)	(1,1)	(2,6)	G_8	7	KC_7	(0,1)	(1,2)	(2,0)
	21	KC_{21}	(0,0)	(1,1)	(2,0)		22	KC_{22}	(0,1)	(1,2)	(2,1)
	36	KC_{36}	(0,0)	(1,1)	(2,1)		37	KC_{37}	(0,1)	(1,2)	(2,2)
	51	KC_{51}	(0,0)	(1,1)	(2,2)		52	KC_{52}	(0,1)	(1,2)	(2,3)
	66	KC_{66}	(0,0)	(1,1)	(2,3)		67	KC_{67}	(0,1)	(1,2)	(2,4)
	81	KC_{81}	(0,0)	(1,1)	(2,4)		82	KC_{82}	(0,1)	(1,2)	(2,5)
	96	KC_{96}	(0,0)	(1,1)	(2,5)		97	KC_{97}	(0,1)	(1,2)	(2,6)
G_9	8	KC_8	(0,2)	(1,3)	(2,1)	G_{10}	9	KC_9	(0,0)	(1,4)	(2,2)
	23	KC_{23}	(0,2)	(1,3)	(2,2)		24	KC_{24}	(0,0)	(1,4)	(2,3)
	38	KC_{38}	(0,2)	(1,3)	(2,3)		39	KC_{39}	(0,0)	(1,4)	(2,4)
	53	KC_{53}	(0,2)	(1,3)	(2,4)		54	KC_{54}	(0,0)	(1,4)	(2,5)
	68	KC_{68}	(0,2)	(1,3)	(2,5)		69	KC_{69}	(0,0)	(1,4)	(2,6)
	83	KC_{83}	(0,2)	(1,3)	(2,6)		84	KC_{84}	(0,0)	(1,4)	(2,0)
	98	KC_{98}	(0,2)	(1,3)	(2,0)		99	KC_{99}	(0,0)	(1,4)	(2,1)

Table 5: Continue

G_n	*id*	KC_{id}	(0,a0)	(1,a1)	(2,a2)	G_n	*id*	KC_{id}	(0,a0)	(1,a1)	(2,a2)
G_{11}	10	KC_{10}	(0,1)	(1,0)	(2,3)	G_{12}	11	KC_{11}	(0,2)	(1,1)	(2,4)
	25	KC_{25}	(0,1)	(1,0)	(2,4)		26	KC_{26}	(0,2)	(1,1)	(2,5)
	40	KC_{40}	(0,1)	(1,0)	(2,5)		41	KC_{41}	(0,2)	(1,1)	(2,6)
	55	KC_{55}	(0,1)	(1,0)	(2,6)		56	KC_{56}	(0,2)	(1,1)	(2,0)
	70	KC_{70}	(0,1)	(1,0)	(2,0)		71	KC_{71}	(0,2)	(1,1)	(2,1)
	85	KC_{85}	(0,1)	(1,0)	(2,1)		86	KC_{86}	(0,2)	(1,1)	(2,2)
	100	KC_{100}	(0,1)	(1,0)	(2,2)		101	KC_{101}	(0,2)	(1,1)	(2,3)
G_{13}	12	KC_{12}	(0,0)	(1,2)	(2,5)	G_{14}	13	KC_{13}	(0,1)	(1,3)	(2,6)
	27	KC_{27}	(0,0)	(1,2)	(2,6)		28	KC_{28}	(0,1)	(1,3)	(2,0)
	42	KC_{42}	(0,0)	(1,2)	(2,0)		43	KC_{43}	(0,1)	(1,3)	(2,1)
	57	KC_{57}	(0,0)	(1,2)	(2,1)		58	KC_{58}	(0,1)	(1,3)	(2,2)
	72	KC_{72}	(0,0)	(1,2)	(2,2)		73	KC_{73}	(0,1)	(1,3)	(2,3)
	87	KC_{87}	(0,0)	(1,2)	(2,3)		88	KC_{88}	(0,1)	(1,3)	(2,4)
	102	KC_{102}	(0,0)	(1,2)	(2,4)		103	KC_{103}	(0,1)	(1,3)	(2,5)
G_{15}	14	KC_{14}	(0,2)	(1,4)	(2,0)						
	29	KC_{29}	(0,2)	(1,4)	(2,1)						
	44	KC_{44}	(0,2)	(1,4)	(2,2)						
	59	KC_{59}	(0,2)	(1,4)	(2,3)						
	74	KC_{74}	(0,2)	(1,4)	(2,4)						
	89	KC_{89}	(0,2)	(1,4)	(2,5)						
	104	KC_{104}	(0,2)	(1,4)	(2,6)						

Table 6: Key chain and its complement key chain

id	(0,a0)	(1,a1)	(2,a2)	Mapping	*id'*	(0,a0')	(1,a1')	(2,a2')
0	(0,0)	(1,0)	(2,0)		104	(0,2)	(1,4)	(2,6)
1	(0,1)	(1,1)	(2,1)		103	(0,1)	(1,3)	(2,5)
2	(0,2)	(1,2)	(2,2)		102	(0,0)	(1,2)	(2,4)
3	(0,0)	(1,3)	(2,3)		101	(0,2)	(1,1)	(2,3)
4	(0,1)	(1,4)	(2,4)		100	(0,1)	(1,0)	(2,2)
5	(0,2)	(1,0)	(2,5)		99	(0,0)	(1,4)	(2,1)
6	(0,0)	(1,1)	(2,6)		98	(0,2)	(1,3)	(2,0)
7	(0,1)	(1,2)	(2,0)		97	(0,1)	(1,2)	(2,6)
8	(0,2)	(1,3)	(2,1)	⟺	96	(0,0)	(1,1)	(2,5)
9	(0,0)	(1,4)	(2,2)		95	(0,2)	(1,0)	(2,4)
10	(0,1)	(1,0)	(2,3)		94	(0,1)	(1,4)	(2,3)
11	(0,2)	(1,1)	(2,4)		93	(0,0)	(1,3)	(2,2)
12	(0,0)	(1,2)	(2,5)		92	(0,2)	(1,2)	(2,1)
13	(0,1)	(1,3)	(2,6)		91	(0,1)	(1,1)	(2,0)
14	(0,2)	(1,4)	(2,0)		90	(0,0)	(1,0)	(2,6)
15	(0,0)	(1,0)	(2,1)		89	(0,2)	(1,4)	(2,5)
16	(0,1)	(1,1)	(2,2)		88	(0,1)	(1,3)	(2,4)
17	(0,2)	(1,2)	(2,3)		87	(0,0)	(1,2)	(2,3)
18	(0,0)	(1,3)	(2,4)		86	(0,2)	(1,1)	(2,2)
19	(0,1)	(1,4)	(2,5)		85	(0,1)	(1,0)	(2,1)
20	(0,2)	(1,0)	(2,6)		84	(0,0)	(1,4)	(2,0)
21	(0,0)	(1,1)	(2,0)		83	(0,2)	(1,3)	(2,6)
22	(0,1)	(1,2)	(2,1)		82	(0,1)	(1,2)	(2,5)
23	(0,2)	(1,3)	(2,2)		81	(0,0)	(1,1)	(2,4)
24	(0,0)	(1,4)	(2,3)		80	(0,2)	(1,0)	(2,3)
25	(0,1)	(1,0)	(2,4)		79	(0,1)	(1,4)	(2,2)
26	(0,2)	(1,1)	(2,5)	⟺	78	(0,0)	(1,3)	(2,1)
27	(0,0)	(1,2)	(2,6)		77	(0,2)	(1,2)	(2,0)
28	(0,1)	(1,3)	(2,0)		76	(0,1)	(1,1)	(2,6)
29	(0,2)	(1,4)	(2,1)		75	(0,0)	(1,0)	(2,5)
30	(0,0)	(1,0)	(2,2)		74	(0,2)	(1,4)	(2,4)
31	(0,1)	(1,1)	(2,3)		73	(0,1)	(1,3)	(2,3)
32	(0,2)	(1,2)	(2,4)		72	(0,0)	(1,2)	(2,2)
33	(0,0)	(1,3)	(2,5)		71	(0,2)	(1,1)	(2,1)
34	(0,1)	(1,4)	(2,6)		70	(0,1)	(1,0)	(2,0)
35	(0,2)	(1,0)	(2,0)		69	(0,0)	(1,4)	(2,6)
36	(0,0)	(1,1)	(2,1)		68	(0,2)	(1,3)	(2,5)

two, the number of common keys that exists among the members of a group is two.

Complementary design: Given a MKPDP with a parameter of n+1 PRP numbers m_i of modulo M such that $N \leq M/2$, where $M=\prod_{i=0}^{n} m_i$. The Key Chain set ($Z_{KC_{id}}$) is derived from the node identifier *id* as stated in Eq. (5).

Then, the Complementary design has the complementary Modulo Key Chain set ($Z'_{KC_{id}}$), which is derived from the key chain set ($Z_{KC_{id}}$) by using the Eq. (9) and (10):

$$KC_{id'} = KC_{((M-1)-id)} \quad (9)$$

$$(i, a'_i) = (i, (m_i - 1) - a_i) \quad (10)$$

If $(Z_{KC_{id}})$follows the Modulo KPD, then $(Z'_{KC_{id}})$ is also follows the Modulo KPD. This complementary property of the Modulo KPD leads to following findings listed as below.

Corollary 1: If M is even, tuples generated from 0 to M/2-1 are complement of M/2 to M-1 and vice versa.

Corollary 2: If M is odd, tuples generated from 0 to M/2 are complement of M/2 to M-1 and vice versa.

Corollary 3: Key connectivity of the M tuples is equivalent to M/2 tuples if M is even and M/2+1 tuples if M is odd.

This complementary property opens a door to increase the scalability of the network. Select the PRP numbers such that (M/2>network size). Then this M/2 tuples are loaded in to the sensor nodes and remaining M/2 tuples can be used for future sensor nodes, which belong to the network.

Here, the Table 6 shows the key chain and its complement key chain of example 2.

CONCLUSION

The pair wise key pre-distribution is used to provide secure communication among sensor nodes deployed in hostile environment. In this study, we have given the overview of the basic modulo scheme. Here we have presented an objective for maximizing the key connectivity among the sensor nodes by enhancing the basic modulo scheme. Modulo function is used here for generating the key chains and for link establishment, modulo function trade is utilized which uses the key overlapping concept. The trade property of modulo function provides a special case of group key pre distribution scheme where different ways of groupings are possible. This modulo function trade achieves full key connectivity among the sensor nodes of each sub group. This trade property of MKPDP is used to apply for the group communication is explained. This MKPDP comprises a complementary property, which leads to high scalability for the network is also explained. Since this MKPDP makes use of small key pool size, it provides low resilience when the number of nodes captured is increased.

REFERENCES

Akyildiz, I.F., W. Su, Y. Sankarasubramaniam and E. Cayirci, 2002. Wireless sensor networks: A survey. Comput. Netw., 38: 393-422.

Bechkit, W., Y. Challal, A. Bouabdallah and A. Bencheikh, 2010. An efficient and highly resilient key management scheme for wireless sensor networks. Proceeding of IEEE 35th Conference on Local Computer Networks (LCN, 2010), pp: 216-219.

Eschenauer, L. and V.D. Gligor, 2002. A key-management scheme for distributed sensor networks. Proceeding of 9th ACM Conference on Computer and Communications Security, pp: 41-47.

Kausar, F., S. Hussain, L.T. Yang and A. Masood, 2008. Scalable and efficient key management for heterogeneous sensor networks. J. Supercomput., 45: 44-65.

Kavitha, T. and D. Sridharan, 2011. A novel identity based deterministic key distribution for wireless sensor network. Eur. J. Sci. Res., 54(3): 363-374.

Kavitha, T. and D. Sridharan, 2013a. Probabilistic key chain based key distribution schemes for WSN. Int. Rev. Comput. Software, 8(5): 1156-1169.

Kavitha, T. and D. Sridharan, 2013b. Key distribution scheme using modulo operation for WSN. Information, 16(11): 8213-8228.

Lee, J.C., K.H. Wong, J. Cao, H.C.B. Chan and V.C.M. Leung, 2007. Key management issues in wireless sensor networks: Current proposals and future developments. IEEE Wirel. Commun., 14(5): 76-84.

Levi, A., S.E. Taşçı, Y.J. Lee, Y.J. Lee and E. Bayramoğlu and M. Ergun, 2010. Simple, extensible and flexible random key predistribution schemes for wireless sensor networks using reusable key pools. J. Intell. Manuf., 21(5): 635-645.

Mehta, M., D. Huang and L. Harn, 2005. RINK-RKP: A scheme for key predistribution and shared-Key discovery in sensor networks. Proceeding of the 24th IEEE International Conference on Performance Computing and Communication (IPCCC'05), pp: 193-197.

Pietro, R.D., L.V. Mancini and A. Mei, 2003. Random key-assignment for secure wireless sensor networks. Proceeding of 1st ACM Workshop Security of Ad Hoc and Sensor Networks (SASN '03). ACM Press, New York, pp: 62-71.

Po-Jen, C., C. Tun-Hao and L. Bo-Yi, 2005. A scalable grouping random key predistribution scheme for large scale distributed sensor networks. Proceedings of the 3rd International Conference on Information Technology and Applications (ICITA'05), 2: 535-540.

Pottie, G.J. and W.J. Kaiser, 2000. Wireless integrated network sensors. Commun. ACM, 43: 551-558.

Ren, K., K. Zeng and W. Lou, 2006. A new approach for random key pre-distribution in large-scale wireless sensor networks. Wirel. Commun. Mob. Com., 6(3): 307-318.

Ruj, S., A. Nayak and I. Stojmenovic, 2011. Fully secure pairwise and triple key distribution in wireless sensor networks using combinatorial designs. Proceedings of the IEEE INFOCOM, pp: 326-330.

Shan, T.H. and C.M. Liu, 2008. Enhancing the key pre-distribution scheme on wireless sensor networks. Proceeding of IEEE Asia-Pacific Services Computing Conference, pp: 1127-1131.

William, S., 2013. Cryptography and Network Security. 3rd Edn., Pearson Education, 2013.

Zeen, K., J. Kim and K. Kim, 2007. Key pre-distribution scheme for wireless sensor networks with higher connectivity. Proceeding of Symposium on Cryptography and Information Security Sasebo. The Institute of Electronics, Information and Communication Engineers, Japan, Jan. 23-26.