Two Approaches on Pairwise Key Path Establishment for Sensor Networks*

Ping Li[1], Yaping Lin[2], Jiaying Wu[1]

[1]School of Computer and Telecommunications, Changsha university of Science and Technology, 410076, Changsha, Hunan, China
[2]College of Software, Hunan University, 410072, Changsha, Hunan, China

{lping9188@163.com, yplin@hnu.cn, jiayin528@163.com }

Abstract. Developments on WSN technologies have made applications of ubiquitous computing available in recent a few years. The properties of weak connectivity in subsection based on hypercube model are addressed, for purpose of achieving inner-area pairwise key establishment. Also a clustering scheme for inter-area path establishment is proposed, based on the contribution of our presented protocol for key information exchange. Security analysis and performance issue are also addressed.

Keywords: pairwise key, sensor networks, ubiquitious computing, hypercube, security

1. Introduction

More and more research attentions have been attracted on the security issue in wireless sensor networks (WSN) because of their tremendous applications available in military as well as civilian areas[1]. At the same time, rapid development on relative technologies of WSN have made applications of ubiquitous computing available in recent a few years. Security schemes of pairwise key establishment, which enable sensors to communicate with each other securely, play a fundamental role in research on security issue in wireless sensor networks[2].

One of the most important issues on pairwise key establishment is key-pre-distribution phase. Two kinds of pre-distribution schemes are available, centralized and localized schemes according to information pre-loaded in each sensors. With respect to the former, Eeschnaure et al[4] presented a probabilistic key pre-distribution scheme. For each node in this scheme, *m* keys are randomly selected

* Supported by Scientific Research Fund of Hu'nan Provincial Education Department of China under Grant No.06B005

from the key pool S and stored into the node's memory so that any two sensors have a certain probability of sharing at least one common key. Based on the contributions made by [5], a lot of attention has focused on polynomial based key pre-distribution. That is, the key setup server randomly generates a t-degree bivariate polynomial $f(x,y)$ over a finite field F_q. Notes that for any variables x and y, $f(x,y)= f(y,x)$ is always held. For any two nodes i and j, the key server computes two shares of $f(x,y)$, denoted as $f(i,y)$ and $f(j,y)$, for the two nodes respectively. Thus they can compute the common key $f(i,j)$ directly.

However, those approaches have some limitations. For the centralized schemes such as probabilistic and polynomial-based schemes[4][6], a small number of compromised sensors may reveal a large fraction of pairwise keys shared by non-compromised sensors. Polynomial-based scheme can only tolerate no more than t compromised nodes, while the value of t is limited due to the memory constrains of sensor nodes[5].

As security challenges arise in centralized schemes, localized schemes have become research focus. Liu et al[7] developed a general framework of polynomial pool-based key pre-distribution and proposed two instantiations, a random subset assignment scheme and a hypercube-based assignment scheme for key pre-distribution. Liu et al[8] also presented a location-aware deployment model, and developed corresponding pairwise key pre-distribution scheme, using location information. The scheme took advantages of the observation that in static sensor networks, although it is difficult to precisely pinpoint sensors' positions, it is often possible to approximately determine their locations.

One the other hand, the possibility to establish direct key in the hypercube-based assignment is not perfect. Furthermore, [7] makes an assumption that a node's signal range can cover the entire network. That implicitly means all the other nodes in a network are direct neighbors for a given nodes. The location-based scheme can achieve better performance due to the explicit usage of location information, while the polynomial pool is fairly small, as sensors expected in the same area have common polynomial shares.

In this paper, we develop two kinds of pairwise key establishment to address above problems in sensor networks. The contribution of this paper is two-fold. First, we model a local network with densely distributed nodes as a hypercube, inspect properties of k-dimensional weak-connectivity in subsection of hypercube model, and develop an effective scheme on pairwise key establishment for inner-area nodes. The resulting scheme guarantees any two sensors appropriating the connectivity requirements to establish pairwise key even though a large number of nodes are compromised or out of communication. Second, a clustering scheme for inter-area path establishment is proposed, based on the contribution of our presented protocol for key information exchange, focusing on networks of large scale with huge number of nodes in a wide deployment area. Our analysis indicates that presented schemes provide nice performance on isolated connected graph clustering as well as key path establishment in inner-area communications.

The rest of this paper is organized as follows. After a brief description on hypercube-based pairwise key establishment in Section 2, Section 3 inspects properties of k-dimensional weak-connectivity in subsection of hypercube model, and presents inner-area scheme on pairwise key establishment. Section 4 describes a clustering scheme as well as a security protocol for pairwise key path establishment. Section 5 analyzes performance of presented schemes, before Section 6 concludes this paper.

2. Hypercube-Based Pairwise Key Establishment Schemes

Given a total of N sensor nodes in the network, this scheme constructs an n-dimensional hypercube with m^{n-1} bivariate polynomials arranged for each dimension j,

$\{ f^{j}_{<i_1,\ldots i_{n-1}>}(x,y) \}_{0 \le} i_1, i_2 .. i_{n-1} < m$, where $m = \lceil \sqrt[n]{N} \rceil$. A node's coordinate is encoded in the hypercube into a single-valued node ID. Every valid coordinate in the hypercube is first converted into n l-bit binary strings (one for each dimension) where $l = \lceil \log_2 m \rceil$. Each ID j is expressed as $<j_1,j_2,\ldots j_n>$, where j_i is called the sub-index of ID j in dimension i, which is also represents the i^{th} l bits of j.

The key setup server randomly generates $n*m^{n-1}$ bivariate t-degree polynomial pool over a finite fields F_q, denoted as

$F=\{ f^{i}_{<i_1,i_2,\ldots,j_{n-1}>}(x,y) \mid 0 \le i_1, i_2, \ldots i_{n-1} \le n, 1 \le j \le n \}$.

For each node, the setup server then selects an unoccupied coordinate $(j_1, j_2, \ldots, j_n)$ in the n-dimensional space and assigns it to this node. The setup server then distributes the following polynomial shares:

$\{ f^{1}_{<j_2,\ldots,j_n>}(x,y),\ f^{2}_{<j_1,j_3,\ldots,j_n>}(x,y), \ldots,\ f^{n}_{<j_1,j_2,\ldots,j_{n-1}>}(x,y) \}$

to this sensor node.

To establish a pairwise key with node j, node i checks whether they have the same sub-index in n-1 dimensions. That is, if both the nodes are logical neighbors in the hypercube, expressed as $d_h(i, j) = 1$, they share a common polynomial, and thus they can establish a direct key. Otherwise, they need to go through path discovery to establish indirect key. If there are no compromised nodes and any two nodes can communicate with each other, the node assignment algorithm guarantees at least one key path that can be used to establish a session key between any two nodes. Alternative key paths are created by dynamic key path discovery in case that intermediate nodes have been compromised. Please refer to [7] for details.

3. Inner-area Pairwise Key Path Establishment

In this Section we consider a simpler situation: Assume that a fairly large number of sensor nodes are densely distributed in a small area. We believe that only in that kind of situation does it make sense to apply the properties of basic hypercube model on pairwise key establishment. In this Section, we mainly inspect the weak connectivity issue on hypercube, and present a scheme on inner-area pairwise key establishment.

3.1 Weak Connectivity Model in Subsection

Consider an n-dimensional hypercube with a total of N sensor nodes, and each node in the network is assigned to a unique coordinate $j_1 j_2 ... j_n$, where $0 \le j_1,...j_n < v$ and $v = \lceil \sqrt[n]{N} \rceil$. For simplicity that n-dimensional hypercube can be expressed as $H(v,n)$. In addition, every valid coordinate can be divided into r subsections in sequence, each of which has no more than k characters, where $r = \lceil n/k \rceil$.

Definition 1: The nodes A and B in $H(v,n)$ are called logic neighbors, iff that only one character is different in their coordinates (as indicated in **Requirement1**). Both the two nodes are called physical neighbors, iff that they are within each other's signal range (as indicated in ***Requirement2***, where d_r denotes node's signal range). There exists a secure link between A and B if they are neighboring both logically and physically.

Requirement1: $d_h(A,B) = 1$ （1）

Requirement2: $d_e(A,B) \le d_r$ （2）

Definition 2: The nodes A and B in $H(v,n)$ are called logic neighbors in subsection, iff that only one section has different characters in their coordinates.

Definition 3: For a given character string with the length of n-k, $b_1 b_2 \cdots b_{n-k}$, the corresponding k-dimensional hypercube $H(k)$ contains v^k nodes and can be expressed as $b_1\, b_2 \ldots b_{n-k}* \ldots *$, where * denotes a character within 0,…,v-1.

Definition 4: (k-dimensional weak-connectivity in subsection): The hypercube $H(v,n)$ is k-dimensional weak-connected in subsection, if the number of all reachable nodes in each section is larger than $v^k/2$.

Lemma 1: Let an n-dimensional Hypercube $H(v,n)$ satisfies the conditions of k-dimensional weak-connectivity in subsection. Then all the reachable nodes form a connected graph in any two k-dimensional sub-hypercube neighboring in subsection.

Proof. Assume that $r = \lceil n/k \rceil$, a coordinate of the length n is then divided into r sub-strings with the length of no more than k. Let H_k and H'_k are two subsection neighboring k-dimensional sub-hypercubes, expressed as $H_1(k) = b_1 b_2 ... b_{(r'-1)k} \alpha_1 .. \alpha_k b_{(r'+1)k} ... b_{n-k} *...*$ and $H_2(k) = b_1 b_2 ... b_{(r'-1)k} \beta_1 .. \beta_k b_{(r'+1)k} ... b_{n-k} *...*$. Assume that the nodes u and v are reachable nodes, which belong to $H_1(k)$ and $H_2(k)$ respectively. Let $u = b_1 b_2 .. \alpha_1 ... \alpha_k ... b_{n-k} x_1 x_2 ... x_k$ and $v = b_1 b_2 .. \beta_1 ... \beta_k ... b_{n-k} y_1 y_2 ... y_k$. According to the property of k-dimensional weak-connectivity in subsection, there exist a reachable u_1 node in $H_1(k)$ and a reachable v_1 node in $H_2(k)$, expressed as $u_1 = b_1 b_2 .. \alpha_1 ... \alpha_k ... b_{n-k} c_1 c_2 ... c_k$, and $v_1 = b_1 b_2 .. \beta_1 ... \beta_k ... b_{n-k} c_1 c_2 ... c_k$. Note that the two nodes u_1 and v_1 are reachable and they both belong the sub-hypercube $H_c(k)$, denoted as $H_c(k) = b_1 b_2 ... b_{(r'-1)k} *...* b_{(r'+1)k} ... b_{n-k} c_1 .. c_k$. As the sub-hypercube $H_c(k)$ satisfies the conditions of k-dimensional weak-connectivity in subsection, there exists a reachable node *c*, expressed as $c = b_1b_2...b_{(r'-1)k}d_1...d_kb_{(r'+1)k}...b_{n-k}c_1..c_k$. The nodes u_1, v_1 and *c* are connected in $H_c(k)$. Thus the nodes *u* and *v* are connected.

Lemma 2: All of the reachable nodes in *n*-dimensional Hypercube *H*(*v*,*n*), which satisfies the conditions of *k*-dimensional local-weak-connectivity in subsection, form a connected graph.

Proof. For simplicity, we express a valid coordinate of a node with the length of *n* as a string containing *t* characters. Assume that the nodes $u = a_1a_2...a_r$ and $v = b_1b_2..b_r$ where $a_i, b_i \in [0, kv-1]$. With regard to a subsection $a_i, b_i, i \in [0, r]$, there exists a subsection c_i, which enables the nodes *u* and u_1 connected in $H_i(k)$ where $u_1 = a_1..a_{i-1}c_ia_{i+1}...a_r$ and $H_i(k) = a_1a_{i-1} * a_{i+1}...a_r$. Also it makes the nodes *v* and $v_1 = b_1..b_{i-1}c_ib_{i+1}...b_r$ connected in $H_i'(k) = b_1b_{i-1} * b_{i+1}...b_r$. Thus along with no more than 2*r*-1 intermediate nodes, such as $u_1, u_2...u_{r-1}, c = c_1c_2...c_r, v_1, v_2...v_{r-1}$, the nodes *u* and *v* are connected.

3.2 Indirect Key Establishment

Assume that the two physically neighboring nodes A $(i_1, i_2, ..., i_n)$ and B $(j_1, j_2, ..., j_n)$ want to establish pairwise key between them. In case that $d_h((i_1, i_2, ..., i_n), (j_1, j_2, ..., j_n)) = k > 1$, the nodes perform the algorithm on indirect key establishment called ***Inter-Area(S,D)*** illustrated as follows.

The algorithm assumes that during the deployment phase nodes are required to exchange their connectivity information in subsection. That is, every node maintains a table *T* to record reachable nodes in each subsection. Let $S(a_1…a_n)$ and $D(b_1…b_n)$ be the two physically neighboring nodes. As the assumption of the algorithm has addressed, they exchange with each other their connection information in each subsection. Assume that the source node *S* initiates the key establishment phase. It thus creates a temporary table called T_D to record reachable nodes in common subsection code with those of node *D*. Node *S* then performs the following procedures:

P1: The subsection process on node’s coordinate, such as $a_1…a_n \longrightarrow a_1', a_2'...a_r'$, $b_1…b_n \longrightarrow b_1', b_2'...b_r'$ where a'_j, $b'_j \in [0,kv-1]$.

P2: Node *S* maintains a set of different subsections from the node *D*, denoted as $\ell = \{d_1, d_2, ..d_w\}_{d1<d2<..<dw}$, and a path list *P* to recode the constructed key path.

P3: Node *S* checks T_D to find if there exist available nodes in each different subsection. If no available nodes in one of the subsections, the algorithm terminates. Otherwise it goes on the next procedure.

P4: Initialize the path *P*: $P \leftarrow S$ and the temporary node *C* $(c_1c_2…c_r)$: $C = \leftarrow S$.

```
P5: FOR ($i$=1; $i \le w$; $i$++){
        IF ($c_i \neq b'_i$) {
            Search a reachable node in the $i^{th}$ subsection: $C_i = c_1…c_{i-1}x_i\ c_{j+1}…c_r$ in
               the table $T_D$;
            Add the intermediate nodes along with the path between $C$ ($c_1c_2\ldots c_r$)and $C_i$ ( $c_1\ldots c_{i-1}x_i\ c_{j+1}\ldots c_r$ ) to $P$ in sequence;
            }
        $C$ ($c_1c_2\ldots c_r$): $C= \leftarrow C_i$;
      $P$: $P \leftarrow C$;
    }
```

It’s comparatively reasonable to model a sensor network in a small area as a hypercube with the properties of *k*-dimensional weak-connectivity in subsection. As one of necessary requirements, at least $(2^{k-1}+1)^r$ nodes are to be reachable for a network containing 2^n nodes. Let us consider a network with *N*=500 and *k*=*r*=3. If the deployment area can be divided 8 sub-areas, each of which owns 8 groups, a reachable nodes is required to be connected at least 5 areas and 25 groups. As sensors are assumed densely distributed in a small area, we believe that such requirements can be satisfied normally.

4. A Clustering Scheme for Inter-area Key Path Establishment

In case of a sensor network deployed in a large deployment field with a huge number of sensors, it is not reasonable to apply the properties of *k*-dimensional weak-connectivity in subsection to construct a key path throughout the network. Fig.1a shows that there exist a large number of isolated connected graphs in a situation of *N*=4000. Even though some graphs may be overlapped physically, no key path available to connect them with each other. Furthermore, it should never be ignored that a fairly large number of isolated single nodes are available ,as described in Fig.1b. In this section, we present a security protocol and a corresponding clustering algorithm, aiming at achieving further clustering by establishing a secure key path among those isolated graphs.

4.1 A Security Protocol for Pairwise Key Establishment

We denote those isolated connected graphs in Fig.1a as $G_1(V_1,E_1), G_2(V_2,E_2)....G_k(V_k,E_k)$. We assume that nodes *u* and *v* are located in $G_1(V_1,E_1), G_2(V_2,E_2)$ respectively. That is, $u \in V_1$ and $v \in V_2$. We also assume that the two nodes are located within each other’s signal range. Either of these two sensors are required to broadcast a request message with their IDs. Thus nodes *u* and *v* can know if logical neighboring nodes of the peers are available in its connected graph. Without loss of generality, we assume *u* has been sure that there exist a node in V_1, u_m, is a logical neighboring node of *v*.The security protocol is presented as follows.

P1: $u \to u_1 \to u_2 .... \to u_m : M$

Within a connected graph $G_1(V_1,E_1)$ node *u* can achieve a key path by performing the algorithm ***Inter-Area(S,D)*** addressed in Section3. A sequence of sensors

$u_1, u_2, \ldots u_m$, that satisfy the requirements (1) and (2), form a key path for *u* to establish a pairwise key with node u_m.

Node *u* generates a random number *r* and initializes an arbitrary sequence number *seq*. It constructs a message $M = \{u, v, d, \{r, seq\}_{k_{temp}}\}$. Here *d* denotes the d^{th} different sub-index between nodes u_m and *v*, k_{temp} denotes currently used pairwise key, such as k_{u,u_1} for node *u* and its next hop u_1. Consider node u_1 having the message $M = \{u, v, d, \{r, seq\}_{k_{temp}}\}$. It decrypts the information and then encrypts it by using the pairwise key shared with its next hop such as u_2. Similar procedures are performed by the following nodes until the message has been transmitted to the destination, u_m.

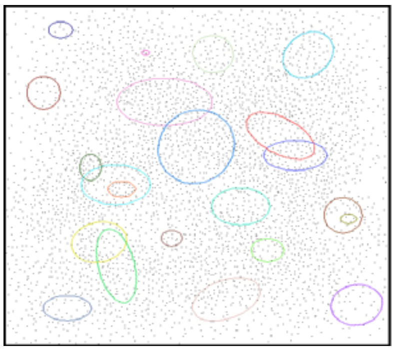
Fig.1a): Isolated connected graphs are available in hypercube-based scheme

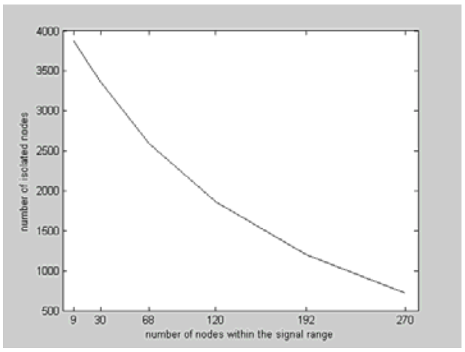

Fig.1b): Relationship between number of isolated nodes and node distribution density given N=4000

P2: $u_m \rightarrow u : \{u, v, d, \{M'\}_{k_p}, \{r_1, ack\}_{k_u}, MAC\}$

When node u_m receives the message *M*, it performs the following processions. First, it calculates $k_u = F(r, seq)$ and encrypts $\{r_1, ack\}$. Here, *F* is a pseudo random function, r_1 is a random number generated by itself. The field *ack* is an acknowledgement corresponding to *seq* such that $ack = f(seq)$, for the received message. Then it chooses the specific polynomial share $f_{u_m}^{d}(u_m, y)$ to create the pairwise key between nodes u_m and *v*, expressed as $k_p = f_{u_m}^{d}(u_m, v)$. Notes that it is used for encryption on the message $M' = \{u, v, ack, r_1\}$. Finally it creates the message authentication code $MAC = C_{k_u}(u, v, d, \{M'\}_{k_p}, \{r_1, ack\}_{k_u})$ and forwards them to node *u*.

P3: $u \rightarrow v : \{u, v, d, \{M'\}_{k_p}\}$

After receiving the message, node *u* performs the following procedures.

Verfication1: Confirm that $D_{k_u}(MAC) = h(u, v, d, \{M'\}_{k_p}, \{r_1, ack\}_{k_u})$

Verification2: Decrypt $\{r_1, ack\}_{k_u}$ to recover r_1 and *ack*. Confirm that $ack = f(syn)$.

Notes that node *u* also has ability to achieve the pairwise key k_u*=F(r,syn)*. By performing above verifications, it authenticates *MAC* to make sure that the reply, which aims at the specific message *M*, is actually originated by node u_m. It then forwards the message $\{u,v,d,\{M'\}_{k_p}\}$ to node *v* directly.

P4: $v \rightarrow u:\{ack,r_v\}_{k'}$

After receiving the messages described above, node *v* first performs the consistency check based on established pairwise key by using the selected polynomial share $f_v^d(v,y)$. It then calculates a session key *k'=F(*r_1*,c),*where *F* is a pseudo random function. Finally, it generates a random number r_v and transmits the message $\{ack,r_v\}_{k'}$ back to node *u.*

P5: $u \rightarrow v:\{r_v\}_{k'}$

Once node *u* has received the message, it creates a session key *k'=F(*r_1*,c)* as it has gained r_1 from node u_m . Then it performs the following verification.

Verification3: Decrypt $\{ack,r_v\}_{k'}$ to recover r_v and *ack*. Confirm that $ack = f(syn)$.

If the verification is successful, it means that the sender is actually the node *v*. In addition, the reply from the node *v* is also validated as it responses properly for node *u*'s request.

Node *v* then performs the following verification to check if node *u* has sent back the correct r_v, which is used for pairwise key establishment.

Verification4: Confirm that $D_{k'}(\{r_v\}_{k'}) = r_v$

The procedure is illustrated as follows.

P1: $u \rightarrow u_1 \rightarrow u_2 .... \rightarrow u_m : M$

P2: $u_m \rightarrow u:\{u,v,d,\{M'\}_{k_p},\{r_1,ack\}_{k_u},MAC\}$

P3: $u \rightarrow v:\{u,v,d,\{M'\}_{k_p}\}$

P4: $v \rightarrow u:\{ack,r_v\}_{k'}$

P5: $u \rightarrow v:\{r_v\}_{k'}$

4.2 Security Analysis

Our protocol can be divided into three phases: 1). Intermediate indirect key establishment phase. 2). Authorized information exchange phase. 3). Key path establishment phase. With regard to the first one, a secure key path is established, along with which any two adjacent nodes share a direct key. Thus, sensible information such as *r* is invisible to any other parties even though they do hear the communications.

The second phase is the heart of the protocol, as it concerns with authentication issue among three parties, such as node *u*, u_m and *v*. With regard to node *u*, **Verification1** and **Verification2** are necessary as it wants to make sure that the reply is actually originated by node u_m. On the other hand, node u_m generates r_1, which enables the other two nodes to create a session key individually. In addition, node u_m uses k_u and k_p to encrypt messages separately, thus achieving node authentication during message exchange. Notes that k_u and k_p are only expected to be created by the designated parties.

The focus of the third phase is the authentication issue on communications between nodes *u* and *v*. Notes that node *v* receives a message with no attachments, it has to initiates a challenge, r_v to the peer node as well as to send back a proper *ack*. By performing **Verification3**, node *u* has a proof to confirm node *v*'s identification. In addition, based on **Verification4**, node *v* is sure that there exists trust relationship between nodes *u* and u_m. Any other node would fail to response the challenge successfully as it has no means to achieve *k'*.

Also, mechanisms such as fault tolerance have been taken into serious consideration. In order to prevent message exchange from network failures, sequence number is introduced to identify a specific round of key path establishment. Any other replies would be discarded if the containing *ack* does not corresponding to a current *seq*.

4.3 A Clustering Algorithm on Inter-Area Key Path Establishment

Consider the two nodes *u* and *v* such that $u \in V_1$, $v \in V_2$, and $d_e(u,v) \le d_r$. Obviously, $d_h(u,v) = k > 1$ is held. Based on the contributions of exchange protocol described in Section 4.1, those two nodes that satisfy the following requirements can establish pairwise key.

Requirement3: $\exists w_1 \in G_1$, $d_h(w_1,v) = 1$ *or*

$$\exists w_2 \in G_2,\quad d_h(u,w_2) = 1 \qquad (3)$$

Thus, those isolated graphs can be clustered further by means of the following algorithm, for purpose of achieving global connectivity through the entire network.

Graph Clustering Algorithm($G_1, G_2, \ldots G_k$**):**

```
P1:   Initialize temporary variables and data structures.
P2:   For (i=1;i<=k;i++)
        {
          If $G_i$ has never been clustered {
               j=j+1;
             $G_j' = G_i$;
              For (l=i+1;l<=k;l++)
              {
                if (Requirement3 can be satisfied)
                   $G_j' = G_j' \cup G_l$;
    }
P3: Output a set of clustered graphs $G_1', G_2', \ldots G_{k'}'$
```

5. Performance

Average number of hops of a key path in inner-area scheme

Consider two nodes $u = a_1 a_2 ... a_r$ and $v = b_1 b_2 .. b_r$ where $a_i, b_i \in [0, kv-1]$. The probability of $a_i = b_i$ for any $i \in \{1, ..., r\}$ is $1/kv$, and the probability of having exactly *i* different subsection is $p[i] = \frac{r!}{i!(r-i)!} \cdot \frac{1}{(kv)^{r-i}} \cdot (1 - \frac{1}{kv})^i$.

Thus, the average key path length ignoring the factor of signal range can be estimated by $L_l = \sum_{i=1}^{r} (2i-1) p[i]$. As the scheme has addressed, we only concern about $v^k / 2$ connected nodes in each subsection. In the worst situation, those nodes may be connected with each other one by one within the signal range, thus the approximate average number of hops is about $v^k / 4 + 1$. Then the average number of hops in a key path can be expressed as $L_h = (\frac{v^k}{4} + 1) \cdot \sum_{i=1}^{r} (2i-1) p[i]$.

Figure2 shows the relationship between the average number of hops and the number of subsections given different local network sizes. The number of hops drops dramatically as the number of subsections grows. Also we can see the larger the network is, the more the required hops.

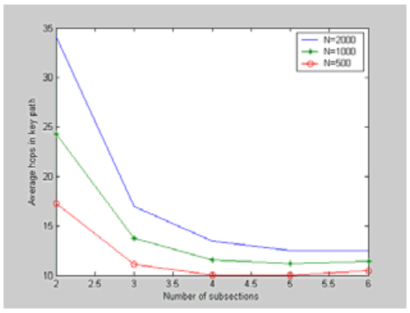

Fig.2: Average number of hops of a key path in inner-area scheme

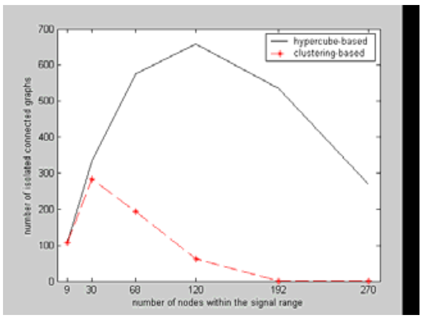

Fig.3: Relationship between number of isolated connected graphs and node distribution density

Probability to achieve graph clustering

Consider two sensors *u* ($u_1 u_2 \ldots u_n$) and $v(v_1 v_2 \ldots v_n)$ where $u \in G_i$ and $v \in G_j$. The probability of having only one different sub-index is $p_1 = n(m-1)/m^n$. Assume that there are N_i and N_j sensors in each area. For a given node *u* (or *v*), the probability of having at least one different sub-index in G_j (or G_i) can be expressed as

$$p_l = (1 - P^0_{N_i - 1}) + (1 - P^0_{N_j - 1}) - (1 - P^0_{N_i - 1}) \cdot (1 - P^0_{N_j - 1}) = 1 - P^0_{N_i - 1} \cdot P^0_{N_{j-1}} = 1 - (1 - p_1)^{N_i + N_j - 2}$$

Fig.3 shows relationship between number of isolated connected graphs and node distribution density given *N*=4000. In case of a network with sparsely distributed nodes, the number of connected graphs is fairly small, as there exist a huge number of “single” nodes. Isolated connected graphs grow dramatically before the node distribution density gets to a certain point. Then number of connected graphs drops as more and more nodes are covered within the signal range. Compared with hypercube-based scheme, our clustering scheme produces much less isolated graphs, converges more quickly on graph clustering, and thus achieves higher probability on global connectivity establishment.

6. Conclusion

Due to energy constraints and random distribution of nodes in sensor networks, we argue that it has some limitations to model connectivity of nodes as a pure hypercube for analysis. We have made two approaches to achieve pairwise key path establishment according to different network sizes. Firstly, we consider a simple situation such as a local network with densely distributed nodes. It is more reasonable to model such a network as a hypercube, and we inspect the connectivity issue in subsection to deal with the situation of a number of nodes are out of communication. Secondly, aiming at the clustering issue on isolated connected graphs in a large target field, we present a security protocol for key path establishment in inter-area communications. Based on the contributions of our protocol, the resulting schemes have nice performance on probability to establish pairwise key through the entire network.

References

1. Chee-yee Chong, Srikanta,P.Kumar. Sensor Networks: Evolution, Opportunities, and Challenges. Proceeding of the IEEE, 2003, 91(8):1247-1256.
2. Du,W., Deng,J., Han,Y.S. et.al. A Pairwise Key Predistribution Scheme for Wireless Sensor Networks, In Proceedings of 10th ACM Conference on Computer and Communication Security. 2003, 42-51.
3. Estrin,D., Govindan,R., Heideman,J., Kumar,S. Next century challenges: Scalable Coordination in Sensor Networks. Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, Pages 263-270, 1999.
4. Eeschnaure,L., Gligor,V.D. A Key-management Scheme for Distributed Sensor Networks, In proceedings of the 9th ACM Coference on Computer and Communication Security. 2002, 41-47.
5. Blundo,C.., Desantis,A., Kutten,S., et.al. Perfectly Secure Key Distribution for Dynamic Conferences. In Advances in Cryptology-CRYPTO’92, 1993, LNCS, 740, 471-486.
6. Chan,H., Oerrig,A., Song,D. Random Key Predistribution Schemes for Sensor Networks, In IEEE Syposium on Research in Security and Privacy. 2003, 197-213.
7. Donggang Liu, Peng Ning, Rongfang Li, Establishing Pairwise Keys in Distributed Sensor Networks. ACM Transactions on Information and System Security. 2005,8(1): 41-77.
8. Donggang Liu, Peng Ning. Location-Based Pairwise Key Establishments for Static Sensor Networks. Available from http://discovery.csc.ncsu.edu/~pning/pubs/sasn03.pdf.