A Plan-Commit-Prove Protocol For Secure Verification Of Traversal Path

Subhas Kumar Ghosh, Amitabha Ghosh

Honeywell Technology Solutions Laboratory
151/1, Doraisanipalya, Bannerghatta Road, Bangalore, INDIA, 560076
{subhas.kumar, amitabha.ghosh }@honeywell.com

***Abstract*— Security is an important issue in wireless sensor networks. With the growing scope of ad-hoc wireless sensor networks, many of its applications require more than just data authentication. The sensor nodes that receive data from other sensor nodes not only need to verify whether the data has been modified on transit but also need to ensure certain associated information with the data. In particular, in this paper we discuss the problem when a mobile node sends data as well as its location information to the static sensor nodes deployed randomly over a two dimensional area. It could be seen, that, a simple data authentication mechanism cannot ensure that the mobile node always provides the correct location information and it does not cheat about the location or the path it has taken for movement. In this paper we provide a "plan-commit-prove" protocol for secure verification of location and traversal path claim and analyze the security aspects of the protocol. The mobile node generates a movement plan and broadcasts a commitment to some neighboring nodes; at a later point in time static nodes verify the planed movement claims of the mobile node by sampling the commitments at various points on the path.**

I. Keywords

Wireless sensor networks, Mobility, Data authentication, Secure location claim verification.

II. Introduction

Emergence of ad-hoc wireless sensor networks has opened up possibilities for a wide range of applications in military and civil areas. Some of these applications like asset tracking, first responder etc, require handling a mix of mobile and static sensor networks.

In most of these applications the current location of the mobile node is important. The mobile node might associate the data it has with its current location so that it is more useful for the application. The application could use the location data for several purposes. One such use could be location based resource management or location aware access control. Such applications need to trust the claimed location of the mobile nodes.

Cryptographic mechanisms like entity or data authentication can ensure that a mobile node can be trusted or the data sent by a mobile node has not been modified in transit. However, it cannot ensure that the mobile node does not claim its presence at a wrong location, thus cheating.

In this paper we define a protocol, where by a mobile node can plan its movement in a pre-defined area; commit its plan to the static nodes and in future prove that indeed it is meeting its planned movements. In section 3 we introduce the model of movement on a sensor-augmented environment and in section 4 we formally introduce the location claim problem. In section 5 we describe the plan-commit-prove protocol. In section 6 we compare with related work and in section 7 we provide an analysis of the protocol in terms of its efficiency and bound.

III. Deployment Scenario

In this section, we first define the sensor placement model in a two dimensional area. Next, we describe the mobility model and the stochastic process that the sensor nodes would follow in terms of random pairwise distances between two nodes and trajectory of the transmitter.

A. Sensor placement model

Ad-hoc wireless sensor networks are formed by groups of sensor nodes that communicate with each other through wireless channels in a multihop fashion. They operate in a decentralized and self-organizing manner and do not rely on any fixed network infrastructure. This necessitates that such an ad-hoc network should be fully connected and there must be multihop paths from each node to another.

B. Mobility Model

Once the sensor nodes are uniformly distributed in the terrain the transmitter node makes a tour over this area, possibly pausing at several places. The mobility model that we consider is the Random Waypoint Model (RWP). It is a stochastic model that describes the movement behavior of a mobile node in a given area. In a RWP, a mobile node randomly chooses a destination point ('waypoint') in the area and moves with a constant velocity to that point. Once it reaches there, it pauses for a while, then randomly chooses a next destination and a new velocity and starts moving towards this new destination, and so on. The time taken by the mobile node to move from one starting position to the next destination position is termed as one $epoch$ and the distance between the two positions is termed as $epoch\, length$.

Let us define the following variables for the mobility model:

- P_i denote the Cartesian coordinates of a waypoint that a mobile node will choose at the i^{th} epoch

- $v_{i,i+1}$ denote the velocity that a mobile node will choose for movement between waypoints P_i and P_{i+1}
- T_i denote the pause time at waypoint P_i
- L_i denote the epoch length between the waypoints P_i and P_{i+1}

The complete movement of a node is then given by the sequence $\{(P_i, v_{i,i+1}, T_i)\}_{i \in N}$. The values for the pause time are chosen from a bounded random distribution $f_T(t)$ in the interval $[0, \tau_{max}]$. The velocities are chosen uniformly within the interval $[v_{min}, v_{max}]$ with $v_{min} > 0$. While the random waypoints are independent and identically distributed over the area A, the distances are not stochastically independent because the endpoint of one movement epoch is the starting point of the successive movement epoch.

IV. A Secure Verification Of Traversal Path

We consider mobile sensor nodes move in a predefined area and we would like to provide an efficient cryptographic protocol to prevent the transmitter from making false claim of its location, where efficiency is accounted in terms of communication cost, message overhead and storage requirements in the network. We would also want to verify that the transmitter nodes can plan their movement ahead of time and meet those plans, or it should be possible to determine if they miss their planned movement.

Having the above mentioned sensor placement and mobility model we formally state the verification of traversal path problem as follows. As shown in figure 1 a mobile node plans its tour by choosing a set of future locations i.e., it chooses a finite set of k waypoints $\{P_i,\ i \in k\}$ through which it will pass as it moves from a source to a destination. When the transmitter reaches waypoint P_i, it chooses a velocity $v_{i,i+1}$ uniformly within the interval $[v_{min}, v_{max}]$ with $v_{min} > 0$, and calculates the number of steps m required to reach the next waypoint P_{i+1}.

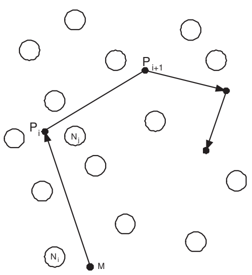

Fig. 1. Movement of mobile nodes in sensor augmented field

The static sensor nodes receive the information from mobile nodes and are expected to receive data from mobile nodes along with their current location. The mobiles nodes though meeting other cryptographic requirements may misinform about their actual locations. A location verification problem can be solved with a location determination capability. However, we are specifically interested in solving it without introducing additional hardware, which might be appropriate for a larger class of sensor network applications.

A. Assumptions

We make the following assumptions:

1) A node trust another node as long as they shares a common key. In this work we also provide a lightweight distributed authentication model without involving the base station or any central authority.
2) Nodes use a pre-distribution mechanism.
3) Existence of one-way function, where given a one-way function $f(\cdot)$ one can generate $k_i = f(k_{i+1})$ such that knowledge of k_i will not compromise k_{i+1}.
4) We use a single symmetric bivariate t-degree polynomial for key establishment, thus given t compromised nodes they can break the complete key-space.
5) Our pre-distribution scheme is based on the work by Blom [5], and later studied by Blundo et.al. in [6] under a bigger framework, and provide the lower bound on the size of the information required in an interactive and non-interactive (t, k) resilient scheme, where t is the size of the group and k is the size of the adversary. Blom's protocol, for a k-secure 2-conference scheme can be thought of as a special case of k-secure t-conference protocol described in [6].
6) We assume that the nodes are time synchronized.

B. Notations

We use the following notations in describing rest of the protocol:

- M is the notation used for mobile node with id m.
- N is the notation used for static sensor node.
- N_x is the notation used for static sensor node with id x.
- $P(x, y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j$ is a symmetric bivariate t-degree polynomial with coefficients a_{ij} over $GF(q)$, where $\binom{t}{2}$ coefficients are unique, and $P(x, y) = P(y, x)$.
- $P(x, y)$ is evaluated at i , as $f_i(y) = P(i, y)$.
- A random location of a mobile node is P_i, and its velocity is $v_{i,i+1}$, when it moves from P_i to P_{i+1}.
- $E\{d\}_{k_{im}}$ is an encryption of data d, under pairwise key k_{im} between node i and m.
- $MAC(msg, \acute{k}_{im})$ is a message authentication code calculated as a digest of message msg under authentication key $\acute{k}_{im}$, which is a pairwise authentication key between node i and m.
- t_0 is the starting time of a key disclosure where key K_0 is disclosed.
- K_i is the key disclosed after the interval $(i \cdot \Delta t + t_0)$.
- $\tilde{y}$ is a multiplicative ϵ-approximation of y or just ϵ approximation if $(1 - \epsilon)y \le \tilde{y} \le (1 + \epsilon)y$.

V. The Plan-Commit-Prove Protocol

As we have described in the previous section, that in a pre-defined area in 2-dimensions $A = a \times b$, where a and b being the dimensions of the deployment area, we will deploy the static nodes randomly and uniformly over the whole area, such that they can provide coverage over the whole space of deployment of static sensor nodes. This will ensure that when one or more mobile nodes move in A, they get connectivity throughout.

A. Pre-distribution

First we define a key pre-distribution mechanism where each node is given with a symmetric polynomial $P(x,y)$ of degree t with coefficients over $GF(q)$, by randomly choosing the coefficients of the polynomial from $GF(q)$, then for each node i ,$P(x,y)$ is evaluated at i , as $f_i(y) = P(i,y)$, and $f_i(y)$ is assigned to each node i. The predistribution process starts with selecting a bivariate t-degree polynomial of form $P(x,y) = \sum_{i,j=0}^{t} a_{ij}x^i y^j$, with coefficients a_{ij} over $GF(q)$, having property that each polynomial is symmetric, i.e. $P(x,y) = P(y,x)$. Now, each node with ID i, is assigned one such polynomial, in the $f_i(y) = P(i,y)$ form after evaluating them at the nodes id. If another node j shares the same polynomial as $f_j(y) = P(j,y)$, then it can evaluate it as $f_j(i) = P(j,i)$, and at the same time knowing j has the same polynomial, i can evaluate it as $f_i(j) = P(i,j)$, and since $P(j,i) = p(i,j)$, they share a secret now. All nodes are given a set of one-way function $f(\cdot)$, along with their id f_{id}, and another one-way function $h(\cdot)$, to be used for deriving data authentication key from established pair-wise key.

B. Plan

Starting from a location P_i, a mobile node makes its planned movement to any randomly selected position P_{i+1}, with a randomly chosen velocity $v_{i,i+1}$. According to our mobility model, this velocity remains constant until the node reaches location P_{i+1}. We call this $\{P_{i+1}, v_{i,i+1}\}$ pair a plan.

C. Commit

Only the mobile node knows the plan. Once the mobile node decides to move, it requests for committing the plan. The request is made as a broadcast and nearby static nodes reply to the request. Assuming the near by node is N_i, and mobile node is M, as described in the protocol in figure 2 the mobile node first sends a nonce n_m and its own id m to N_i. N_i then uses the node M's id to evaluate its own polynomial. Then node N_i generates a random number k_{im} from the $GF(q)$ as a pair-wise key to be used between N_i and M. Node N_i also derives a pair-wise authentication key $\acute{k}_{im} = h(k_{im})$ from k_{im}. Node N_i then sends the id of one-way function to be used in this plan by M, encrypted under their common key and also authenticates the message by calculating a message authentication code using $\acute{k}_{im}$. On receipt node M evaluates its own polynomial at node N_i id, and extracts key k_{im} and derives $\acute{k}_{im}$. At then end node M sends $MAC(n_i||i, \acute{k}_{im})$ to complete the symmetric authentication protocol.

The mobile node uses the one-way function received $f_{id}(\cdot)$ to calculate its key commitments. First it concatenates the next planned location and velocity to derive the first key as $K_n = f_{id}(P_{i+1}||v_{i,i+1})$. Then it calculates the number of keys it needs to derive as:

$M \rightarrow N_i$: n_m, m A nonce and its node id to initiate mutual authentication
N_i : Evaluate $f_i(y)$ at $y = m$, where m is node M's id, where i is id of N_i
N_i : k_{im} Generate a random number from $GF(q)$
N_i : $\acute{k}_{im} = h(k_{im})$ derive an authentication key
N_i : $s_{im} = f_i(m)$
N_i : $\gamma_{im} = s_{im} \oplus k_{im}$
$N_i \rightarrow M$: $n_i, i, E\{f_{id}\}_{k_{im}}$,
$MAC\left[n_i||n_m||i||m||\gamma_{im}, \acute{k}_{im}\right]$ where f_{id} is the id of one-way function to be used by M
M : Evaluate $f_m(y)$ at $y = i$, where i is node N_i's id
M : $s_{mi} = f_m(i)$
M : $k_{im} = \gamma_{im} \oplus s_{mi}$
M : $\acute{k}_{im} = h(k_{im})$ derive an authentication key
M : verify MAC
M : decrypt $E\{f_{id}\}_{k_{im}}$ and extract one-way function id
$M \rightarrow N_i$: $MAC(n_i||i, \acute{k}_{im})$
N_i : verify MAC

Fig. 2. Authenticate and receive one-way function

$$n = \frac{||P_{i+1} - P_i||}{||v_{i,i+1}||} \tag{1}$$

where, $\|\cdot\|$ is the Euclidean norm

$$\|P_{i+1} - P_i\| = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} \tag{2}$$

$$\|v_{i,i+1}\| = \sqrt{[v_x]_{i,i+1}^2 + [v_y]_{i,i+1}^2} \tag{3}$$

The mobile node then applies the one-way function to derive the key chain as $K_i = f_{id}(K_{i+1})$ starting from K_n up to K_0. K_0 forms the commitment to the key chain. Unless declared by the mobile node, no one can retrieve K_{i+1} from the knowledge of K_i. But one can easily verify K_{i+1} from the knowledge of K_i, as $K_i = f_{id}(K_{i+1})$. The mobile then sends the commitment as P_0, t_0, K_0, Δt, $MAC\{P_0 \mid\mid t_0 \mid\mid K_0 \mid\mid \Delta t,\ K_{MN}\}$. Figure 3 shows the key chain derived by node M.

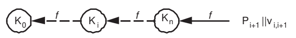

Fig. 3. Key-chain derived by the mobile node

D. Bootstrapping new nodes

However, when a new node gets connected to the mobile node as it moves, it will not be able to verify the received data from the mobile node, as neither it knows about the commitment nor about the one way function in use for the current plan. Observe the scenario in figure 4; initially node N_k is not within the communication range of mobile node M, after a while it gets connected, and when node N_k receives data from M, it needs to verify the data and for that it needs to be bootstrapped. We do it by executing the same two way symmetric authentication and key distribution protocol between new node N_k and mobile node M at the end of which, the current commitment is provided by the mobile node to the new node to be bootstrapped.

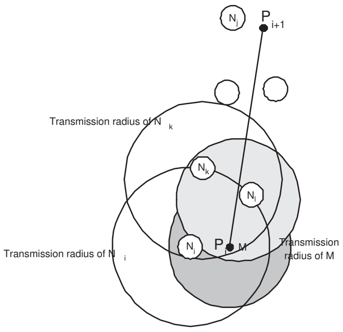

Fig. 4. Bootstrapping new nodes as they come closer to moving node

In figure 5 we describe the protocol for bootstrapping new nodes. The protocol is very similar to the one used for receiving the one-way key. Only difference is that in this protocol, after authenticated key establishment, commitment is provided to node N_k with id of the one-way function by the mobile node M.

$N_k \rightarrow M\ :\ n_k, k$ A nonce and its node id to initiate mutual authentication
$M\ :$ Evaluate $f_m(y)$ at $y = k$, where k is node N_k's id, and m is id of M
$M\ :\ k_{mk}$ Generate a random number from $GF(q)$
$M\ :\ \acute{k}_{mk} = h(k_{mk})$ derive an authentication key
$M\ :\ s_{mk} = f_m(k)$
$M\ :\ \gamma_{mk} = s_{mk} \oplus k_{mk}$
$M \rightarrow N_k\ :\ n_m, m, MAC\left[n_k||n_m||k||m||\gamma_{im}, \acute{k}_{im}\right]$
$N_k\ :$ Evaluate $f_k(y)$ at $y = m$, where m is node M's id
$N_k\ :\ s_{km} = f_k(m)$
$N_k\ :\ k_{mk} = \gamma_{mk} \oplus s_{km}$
$N_k\ :\ \acute{k}_{mk} = h(k_{mk})$ derive an authentication key
$N_k\ :$ verify MAC
$N_k \rightarrow M\ :\ MAC(n_m||m, \acute{k}_{mk})$
$M\ :$ verify MAC
$M \rightarrow N_k : E\left\{f_{id}\right\}_{k_{mk}}, P_i, K_i, t_i, t_0, \Delta t,$
$MAC\left[P_i||K_i||t_i||t_0||\Delta t, \acute{k}_{mk}\right]$
$N_k\ :$ decrypt $E\left\{f_{id}\right\}_{k_{im}}$ and extract one-way function id

Fig. 5. Protocol for bootstrapping new nodes

E. Data Authentication

As the mobile node moves and sends data to the static sensor nodes, it uses the undisclosed key for the current epoch. Sometime after the current epoch is over the mobile node discloses the data authentication key used. The following figure 6 shows the protocol for key disclosure.

Now the recipient node N can verify the received and buffered data as $K_0 = f^{i-1}(K_{i-1})$. This is applicable to the node which has provided the one-way function f to the mobile node and has authenticated it.

$M \rightarrow N:\ K_i,\ i \cdot \Delta t + t_0,$
$MAC\left\{K_i\ ||\ (i \cdot \Delta t + t_0)\ ,\ K_{mn}\right\}$

Fig. 6. Key disclosure

F. Prove

When a mobile node reaches its planned destination P_{i+1}, it will again plan for its next movement, while it need to receive the new one-way function and prove its current location. It will execute the same protocol to authenticate and receive the one-way function id. Since, the static nodes communicate at most with a radius of r_0, it ensures as shown in figure 7, that the mobile node has at least started within the region defined by the transmission range of N_i and ended in N_j. However, it could travel in a different path in the middle. Thus, to complete the proof, the current verifier looks at the previous commitment location P_i, and time t_0, last disclosed location P_{i+1} and time $t_n = n \cdot \Delta t + t_0$ and calculates the velocity of mobile node $v_{i,i+1} = \|P_{i+1} - P_i\|/t_n - t_0$, and $n = t_n - t_0/\Delta t$. Assuming that the mobile node has moved correctly between P_i and P_{i+1}, the prover node now randomly and uniformly selects a pair of epoch i and j as shown in figure 8 and checks the nodes near to location $P_i + i.v_{i,i+1}$ and $P_i + j.v_{i,i+1}$, such that if the nodes near by has received the corresponding K_i and K_j key disclosure was made about time $t_0 + i \cdot \Delta t$ and $t_0 + j \cdot \Delta t$ time, and if $i \leq j$ then $K_i = f^{j-i}(K_j)$, or if $j < i$ then $K_j = f^{i-j}(K_i)$.

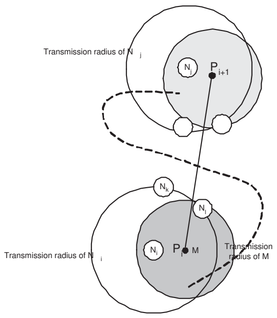

Fig. 7. Proving problem

This interactive proof relies on the fact that the due to the mobility model chosen the nodes around the path from P_i to P_{i+1}, gets connected to mobile node and receives the key disclosure in a perfectly sorted order. Thus we can perform a test using Sort-Check-II spot checker from [8] with subsequent uniform sampling of pairs of elements. The Sort-Check-II spot checker performs $\mathcal{O}((1/c)log(n)log^2(1/\epsilon))$ searches on the sequence to ensure that a $(1-\epsilon)n$ fraction of the elements are sequenced correctly, which results in $\mathcal{O}(log(n/\epsilon))$ and $\mathcal{O}(1/\epsilon)$ samples. We provide the algorithm in figure 9.

VI. Related Works

Among the different mobility models that have been extensively used in simulations and performance evaluations of wireless networks, *random waypoint model* (RWP) is the

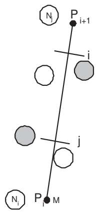

Fig. 8. Spot-checking

Procedure $TraversalCheck(c, \epsilon)$
for $l = 1 \ldots (1/c)log(1/\epsilon)$ **do**
  Choose $i \in_R \{1 \ldots n\}$
  for $k = 0 \ldots log(i)$ **do**
    for $p = 1 \ldots n$ **do**
      Choose $j \in_R \{1 \ldots 2^k\}$
      Request the nodes near to location $P_i + i.v_{i,i+1}$ and $P_i + j.v_{i,i+1}$ for K_i and K_j
      if key disclosure was made about time $t_0 + i \cdot \Delta t$ and $t_0 + j \cdot \Delta t$ time **then**
        if $i \leq j$ **then**
          if $K_i \neq f^{j-i}(K_j)$ **then**
            return FAIL
          end if
        else
          if $K_j \neq f^{i-j}(K_i)$ **then**
            return FAIL
          end if
        end if
      else
        return FAIL
      end if
    end for
  end for
end for
return PASS

Fig. 9. Traversal Check

most common one. Localization algorithms have been studied extensively in the past for accurate location determination of sensor nodes. This include MIT's Cricket location system [10], RADAR [3] by Microsoft Research Group etc. However none of these works addressed security. In a previous work [11] the concept of location claim verification has been introduced and a protocol named *Echo* has been presented for secure location verification. However, their protocol requires extra hardware such as, acoustic hardware capability of the sensor nodes. Echo relies on difference in time-of-flight in RF and sound to estimate the distance of a new node. In this paper we provide data authentication techniques without assuming a trusted base station. We also provide location claim verification for the mobile nodes, which is a very important factor in applications that provide location based access control.

VII. ANALYSIS

A. Security

The objective of the security analysis would be to determine what can be achieved as a bound with plan-commit-prove protocol for any traversal of mobile node between two random points P_i and P_{i+1}, that the mobile node has really traversed from P_i to P_{i+1}. We observer that the disclosure of key will be perfectly sequenced between these two points at Δt intervals. The nodes on this path will be receiving this disclosure in sequence or at most miss one or more key disclosure but the sequence will not change unless the mobile node traverses on a different path. The proof checking algorithm selects the i randomly and uniformly between 1 and n, and performs $\mathcal{O}((1/c)log(n)log^2(1/\epsilon))$ searches on the sequence to ensure that a $(1-\epsilon)n$ fraction of the elements are sequenced correctly, which results in $\mathcal{O}(log(n/\epsilon))$ and $\mathcal{O}(1/\epsilon)$ samples. The results are captured in the following theorem VII.1.

THEOREM VII.1. *Procedure $TraversalCheck(n, \epsilon)$ selects $1/\epsilon$ pairs of number i and j uniformly between $\{1, \ldots, n\}$, runs in time $\mathcal{O}(1/\epsilon)$ and meets the following results: If the mobile node has traversed in straight path between P_i and P_{i+1} with uniform velocity $v_{i,i+1}$ and has disclosed key in sequence, and all nodes are time synchronized, then key K_i has been disclosed after time $(t_0 + i \cdot \Delta t)$ to node closest to location $(P_i + i.v_{i,i+1})$, and key K_j has been disclosed after time $(t_0 + j \cdot \Delta t)$ to node closest to location $(P_j + j.v_{i,i+1})$, and if $i \leq j$ then $K_i = f^{j-i}(K_j)$, or if $j < i$ then $K_j = f^{i-j}(K_i)$, then result is "PASS". If keys were not disclosed by mobile node in sequence or if an increasing subsequence of length $< \epsilon n$ does not get generated in the way mobile node has traversed between P_i and P_{i+1} with uniform velocity $v_{i,i+1}$ and has disclosed keys, then the result is "FAIL".*

PROOF: This directly derives from the Sort-Check-II spot checker, that if the traversal follows the RWP model and the prover moves in straight path between P_i and P_{i+1} with uniform velocity $v_{i,i+1}$, then that will generate a sequence, and by Sort-Check-II, an algorithm can verify by performing $\mathcal{O}((1/c)log(n)log^2(1/\epsilon))$ searches on the sequence to ensure that a $(1-\epsilon)n$ fraction of the elements are sequenced correctly, which results in $\mathcal{O}(log(n/\epsilon))$ and $\mathcal{O}(1/\epsilon)$ samples. ■

B. Storage

Each node needs to store a polynomial for authenticated key establishment. The storage required for this will be $(t+1)log(q)$, where coefficients of the bi-variate polynomial were chosen from $GF(2^q)$. The mobile node needs to calculate the key chain and store. Here we analyze the storage cost of the mobile node. If we consider an area of $20000m \times 10000m$ and velocity uniformly chosen from $[50m/min, 500m/min]$ and a key disclosure every minute, i.e. $\Delta t = 1min$ then the longest distance a mobile node need to traverse will be

$\sqrt{20000 \cdot 20000 + 10000 \cdot 10000} \approx 22361m$, if it chooses the least velocity then the number of keys required will be $22361/50 \approx 450$, if we use 64 bit key then the storage requirement will be $450 \cdot 64 \approx 4kb$.

VIII. Conclusion

In this paper, we have formulated a location and traversal path claim verification problem and provided an algorithm, which ensures within certain bounds that a mobile node would 'really' travel along a predefined path. Our algorithm does not need any extra hardware support on the sensor nodes. Also, instead of assuming a centralized base station, we have provided a lightweight distributed data authentication algorithm to bootstrap any new node, which comes within the transmission radius of the mobile node. Our sensor deployment model is based on a uniform random distribution and the mobility model we have chosen is the very popular *RWP* model. However, it is possible to extend it to any other mobility model as long as it is allowed that mobile node can choose its destination and velocity randomly. We have also provided a security analysis of the traversal path between two successive waypoints and argued that security can be assured unless the mobile node does not travel in the pre-defined path. In other words, if the mobile node traverses in a different path than the pre-defined one, then it's claimed location will not be accepted by the static sensor nodes. In our approach, the mobile node might need to store a maximum of $\frac{\sqrt{a^2+b^2}log(q)}{v_{min}} + (t+1)log(q)$ keys and each of the static sensor nodes will have to store the coefficients of the bi-variate polynomial. The space required for these coefficients is $(t+1)log(q)$. As part of future work, we would try to reduce the storage space.

Acknowledgment

Authors would like to thank the members of ubiquitous computing group at Honeywell Technology Solutions Lab. Author also like to thank Arul Ganesh especially for several discussions and constructive comments that helped improving the paper.

References

[1] The network simulator - ns–2. *http://www.isi.edu/nsnam/ns/*, June 2002.

[2] Glomosim. *http://pcl.cs.ucla.edu/projects/glomosim/*, Nov. 2002.

[3] P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, pages 775–784, 2000.

[4] C. Bettstetter, H. Hartenstein, and X. Pérez-Costa. Stochastic properties of the random waypoint mobility model: epoch length, direction distribution, and cell change rate. In *Proceedings of the 5th ACM international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 7–14. ACM Press, 2002.

[5] R. Blom. An optimal class of symmetric key generation systems. In *Proc. of the EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques*, pages 335–338. Springer-Verlag New York, Inc., 1985.

[6] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 471–486. Springer-Verlag, 1993.

[7] A. Chakrabarti, A. Sabharwal, and B. Aazhang. Using Predictable Observer Mobility for Power Efficient Design of Sensor Networks. In *Information Processing in Sensor Networks*, Palo Alto, CA, April 2003.

[8] F. Ergūn, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 259–268. ACM Press, 1998.

[9] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wirel. Netw.*, 8(5):521–534, 2002.

[10] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM Press, 2000.

[11] N. Sastry, U. Shankar, and D. Wagner. Secure verification of location claims. In *Proceedings of the 2003 ACM workshop on Wireless security*, pages 1–10. ACM Press, 2003.