

Protecting Storage Location Privacy in Sensor Networks

Jianming Zhou¹, Wensheng Zhang¹, and Daji Qiao²

¹Department of Computer Science

²Department of Electrical and Computer Engineering
Iowa State University

Email: {jmzhou,wzhang}@cs.iastate.edu, daji@iastate.edu

November 29, 2006

Abstract

Numerous schemes have been proposed to facilitate data collection and provision in sensor networks, among which the Data-Centric Storage (DCS) scheme is an energy-efficient solution and a popular choice for many sensor network applications. However, since each sensor node in the DCS system knows the locations of all storage nodes, the DCS system is extremely vulnerable to security attacks as a *single* compromised sensor node will expose all the storage locations to the adversary. To address this problem, we propose a *randomized storage concealment scheme* along with a supplementary *storage migration scheme*. In the randomized storage concealment scheme, sensor nodes cooperate to forward data towards the storage nodes without keeping explicit storage locations; instead, each sensor node only maintains the IDs of its randomly-picked next-hop nodes towards the storage nodes. This scheme increases the difficulty significantly for the adversary to derive the storage locations. Nevertheless, the protection provided by this scheme degrades gradually as more and more sensor nodes are compromised. Hence, we further introduce a *storage migration scheme* to supplement the randomized storage concealment scheme, which directs the storage duties to migrate periodically among sensor nodes. Extensive analysis and simulations are conducted to show that the proposed schemes can effectively protect the storage location privacy with modestly added overhead.

1 Introduction

Due to the attractive capabilities of sensing harsh and hostile environments, generating fine-grained sensing data, as well as collecting and provisioning data to remote users, sensor networks have been widely-adopted for many applications [1] such as wild-life monitoring, military target tracking, battlefield surveillance, etc. These applications often require the deployment of large-scale networks to vast areas, which may generate a huge volume of sensing data. This, together with the scarceness in resources and the hostility in operational environments, necessitates the design of efficient and secure data collection and dissemination schemes for large-scale sensor networks.

In recent years, numerous schemes for data collection and dissemination [2–10] have been proposed for sensor networks. One simple approach is to let each sensor node transfer data to a centralized server for long-term archival storage and the server is responsible for servicing queries from users [11]. This approach, however, may incur high network cost for multi-hop data streaming especially when the network scale is large. Alternatively, sensing data may be stored locally at the sensor nodes. To enable users to access the data of their interests, either data sources (i.e., sensor nodes that generate data) or users need to flood their metadata or queries [3, 4]. Although optimization schemes exist to reduce

the cost, such approach could still be very expensive. Lying between the aforementioned two types of approaches is a strategy called *Data-Centric Storage (DCS)* [5]. One typical implementation of DCS is based on *Geographic Hash Table (GHT)* [5], in which each data item is given a name [12] and a hash function is applied on the name to get a location for storing the data. The DCS strategy enables distributed data storage within the network, while allowing direct data query without message flooding. Therefore, it is more efficient than the above two alternatives in many scenarios. The DCS scheme has consequently become a popular choice for many sensor network applications, and various DCS systems [6, 8, 10, 13] have been developed.

Along with the improvement in energy efficiency, the DCS strategy brings new security challenges that have not received adequate attention in the past. Specifically, the DCS scheme requires every sensor node aware of the locations of storage nodes for all data types. Once attackers have captured a single sensor node, it can obtain the data storage locations and may subsequently attack the data-centric storage system by compromising storage nodes or blocking communications between storage nodes and other sensor nodes. Therefore, it is vital to prevent the locations of storage nodes from being exposed to attackers, and to the best of our knowledge, this problem has rarely been addressed before.

In this paper, we propose a *randomized storage concealment scheme* along with a supplementary *storage migration scheme* to protect storage location privacy in the presence of sensor node compromises. In the randomized storage concealment scheme, information relevant to storage locations is blurred in order to increase the difficulty for attackers to infer the locations, which is similar to some prior work [14, 15]. However, one salient difference between our scheme and prior work is that, our scheme is designed for the context of highly distributed and resource-constrained sensor networks; to fit the context, our scheme is designed to be self-organized and *lightweight* based on a novel application of randomization. Specifically, the proposed randomized storage concealment scheme does not require sensor nodes to keep the permanent mapping between data types and storage locations. Instead, each node only keeps partially-perturbed information about the storage nodes, i.e., IDs of next-hop nodes on randomly-selected paths towards the storage nodes. This way, sensing data can still be forwarded to the corresponding storage nodes via cooperation among sensor nodes; meanwhile, due to the randomness in selecting the next-hop nodes, it becomes much more difficult for attackers to derive the storage locations from the routing information captured in compromised sensor nodes. However, it can be expected that the protection provided by this scheme degrades gradually as more and more sensor nodes are compromised; when a significantly large number of sensor nodes have been compromised, attackers may still be able to infer the storage locations. To address such limitation, we further introduce a *storage migration scheme* to supplement the randomized storage concealment scheme, which directs the storage duties to migrate periodically among sensor nodes, and migrations bring forth updates to the partial information stored at individual sensor nodes. Analysis and simulation results verify that the proposed schemes can enhance the storage location privacy significantly with modestly added overhead.

The rest of the paper is organized as follows: Section 2 presents the system model. Section 3 describes, analyzes and evaluates the storage concealment scheme. Section 4 introduces the storage migration scheme, and reports the analysis and evaluation results. The paper concludes in Section 5.

2 System Model

2.1 Network Assumptions

The network assumptions are the same as those of the DCS system [5]. Specifically, we consider a wireless sensor network with a network controller and a large number of sensor nodes deployed to monitor a vast sensing field. These nodes are aware of their own locations using GPS [16] or certain inexpensive localization algorithms such as triangulation [17]. Sensor nodes generate sensing data periodically, and transfer and store them at designated *storage nodes*. Data items are classified according to the application scenario, and data items of the same type are stored at the corresponding storage nodes such that data queries can be directed to the storage nodes instead of being flooded network-wide. The mapping between data types and storage nodes can be implemented by using a GHT-based scheme [5], which uses a one-way hash function H to map each data type to a specific location inside the network, i.e., $H : D \mapsto L$, with D being the set of data types and L being the set of locations. According to the hash function, a data item of type $d \in D$ should be stored at the sensor node closest to the location $H(d) \in L$. All the data types are pre-defined before sensor nodes are deployed. We also assume that the clocks of all sensor nodes are loosely synchronized [18], which is a prerequisite for many distributed sensing applications.

2.2 Security Assumptions

To achieve basic data confidentiality and authenticity, we assume that neighboring sensor nodes can set up pair-wise keys using existing key management schemes such as [19–21]. Each node and its trusted neighbors can establish and maintain a cluster key to secure broadcast within the neighborhood [22]. The network controller is trustworthy and cannot be compromised. Moreover, we also assume that each sensor node is trustworthy before deployment, and it takes non-trivial time to compromise a node; hence in practice, each sensor node is trustworthy within a short period after deployment. In fact, we discuss an enhancement to our proposed scheme in Section 3.3.1, which can remove this assumption if sensor node deployment knowledge is available.

2.3 Attack model

This paper aims at protecting storage location privacy in the DCS system, i.e., preventing attackers from obtaining the storage locations. Generally, the attackers may obtain the storage locations (1) from the relevant information stored at

compromised sensor nodes, or (2) from monitoring the network traffic to find out the traffic pattern (also known as traffic analysis). Our proposed schemes focus on countering the first type of attacks, which has been rarely studied in prior work. In addition, our proposed schemes can also mitigate the second type of attacks, when they are deployed together with previously proposed traffic analysis countermeasures such as [23–26].

3 Storage Concealment

In the GHT-based implementation of the DCS system, each sensor node keeps a hash function H . Therefore, after the adversary compromises a single node, it can obtain the locations of all storage nodes easily and then launch the attacks. Concealing the storage location information on sensor nodes is an attractive idea to thwart this type of attacks, with which each sensor node only keeps *indirect* information about the storage locations. With storage concealment, the adversary needs to compromise a larger number of sensor nodes in order to derive the storage locations, while sensing data can still be forwarded to the corresponding storage nodes via cooperation among sensor nodes.

In this section, we first briefly discuss two preliminary storage concealment schemes, and then describe and evaluate the basic version of our proposed randomized storage concealment scheme in detail. After that, we further propose several enhancements to the basic scheme to achieve a higher level of security, more flexibility in node deployment, and better reliability.

3.1 Preliminary Storage Concealment Schemes

3.1.1 Keeping Directions towards Storage Nodes

To conceal storage locations, a simple extension to the GHT-based implementation of DCS is to let each sensor node keep the directions towards the storage nodes instead of keeping H directly. Specifically, after a node is deployed and has discovered its location, it keeps directions towards all storage nodes computed from H . In practice, for each storage node, the direction associated with it can be represented as an arbitrary point on the line that starts from the sensor node itself and passes through the storage node. This approach, however, cannot tolerate two or more node compromises. As shown in Figure 1(a), after two nodes (A and C) are compromised, the location of the storage node (R) can be derived directly from the directions stored at them.

3.1.2 Keeping Next-hop Nodes on the Shortest Paths towards Storage Nodes

As a further extension to the GHT-based approach, each sensor node may instead keep the next-hop nodes on the shortest paths towards the storage nodes. The idea is detailed as follows: when the sensor nodes are deployed in the sensing field, they discover their neighbors; for each data type, every node obtains the location of the corresponding storage node using

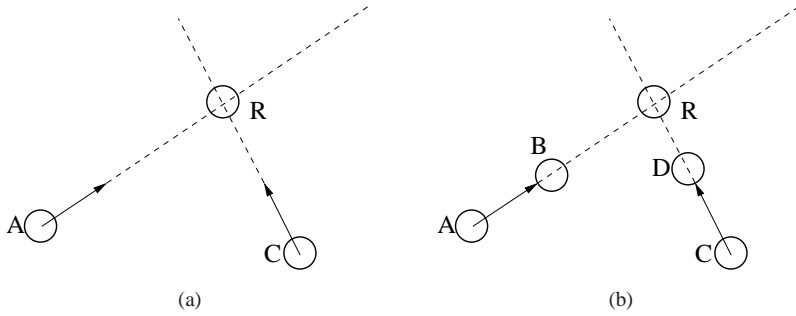


Figure 1: Storage location inference

type	next_hop
001	A
002	B
...	...

Figure 2: An example next-hop routing table

the hash function H , and then chooses the neighbor that is closest to the storage node as the next-hop forwarding node for that data type.

Figure 2 shows an example of the next-hop routing table maintained at each sensor node. Each entry in the table is a pair: $\langle type, next_hop \rangle$, where $type$ is the data type, and $next_hop$ is the ID of next-hop node to forward the data. For each data type, there is only one entry in the table, and the table is empty prior to node deployment.

The transformation from the hash function to the next-hop routing table provides a higher level of storage location privacy than the aforementioned direction-based scheme. However, this approach becomes ineffective when two or more pairs of neighboring nodes are compromised. As illustrated in Figure 1(b), the adversary compromises two pairs of neighboring nodes A, B and C, D, where B and D are the next-hop nodes (towards the storage node R) chosen by A and C, respectively. Note that, when the node density is high, it is very likely that B is close to line AR and D is close to line CR. Thus, the location of R may be approximated accurately by the intersection of AB and CD.

3.2 The Proposed Randomized Storage Concealment Scheme: Basic Version

To thwart the location inference attacks illustrated in Figure 1, we propose a randomized storage concealment scheme that introduces randomness into the procedure of choosing next-hop nodes. In this section, we describe the basic version of randomized storage concealment scheme, which is referred to as *basic scheme* in the rest of this paper, and follow with its data deliverability analysis, communication overhead analysis, security analysis and performance evaluation.

3.2.1 Basic Scheme

After initial deployment, each sensor node discovers its neighbors and chooses next-hop nodes based on the hash function H , and then removes H immediately. However, different from the preliminary scheme in Section 3.1.2, each sensor node does not always choose the neighbor that is closest to the target storage node as the next-hop node; instead, it selects the next-hop node randomly from its neighbors as long as certain requirement is satisfied. The idea is illustrated in Figure 3, where N is the sensor node, R is the target storage node, λ is an angle bisected by line NR, and the dashed circle

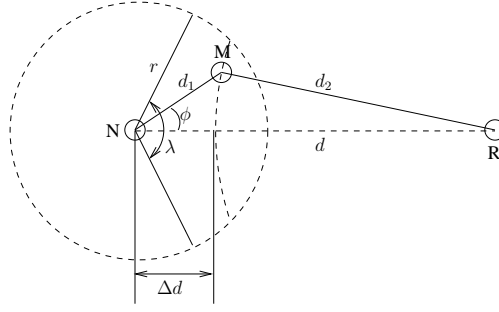


Figure 3: Randomized storage concealment scheme with anonymous angle λ

surrounding N represents the communication range. N randomly picks a neighbor within angle λ as its next-hop node for the storage node R. λ is called the *anonymous angle* and is an important parameter in the basic scheme. Obviously, the larger the λ value, the more randomness is introduced into the scheme, but on the other hand, if λ is larger than a certain value, sensing data can not be guaranteed to reach the target storage node. In the following, we derive the upper bound for λ to guarantee data delivery.

Without loss of generality, consider the sensor node M in Figure 3, where M is picked by N as its next-hop forwarding node towards the target storage node R. Let d_1 , d_2 , and d denote the lengths of NM, MR, and NR, respectively, and let the communication range be r . We have

$$\phi \leq \frac{\lambda}{2} \quad \text{and} \quad d_1 \leq r. \quad (1)$$

To ensure that data from N will eventually reach R, the following condition must be satisfied:

$$d_2 < d. \quad (2)$$

Therefore, for sensor nodes outside the communication range of the target storage node, i.e., $d > r$, we have

$$\begin{aligned} & \forall \phi \leq \frac{\lambda}{2}, \forall d_1 \leq r, \quad d_2 < d \\ \iff & \forall \phi, \forall d_1, \quad d_1^2 + d^2 - 2 \cdot d_1 \cdot d \cdot \cos \phi < d^2 \\ \iff & \forall \phi, \forall d_1, \quad \cos \phi > \frac{d_1}{2d} \\ \iff & \min_{\phi} \cos \phi > \max_{d_1} \frac{d_1}{2d} \\ \iff & \cos \frac{\lambda}{2} \geq \frac{1}{2} \\ \iff & \lambda \leq \frac{2\pi}{3}. \end{aligned} \quad (3)$$

In other words, the valid range of λ is $[0, \frac{2\pi}{3}]$, which is enforced in the basic scheme.

For sensor nodes within the communication range of the target storage node, i.e., $d \leq r$, we adopt a one-hop *point-to-me* message mechanism. Concretely, the storage node sends a *point-to-me* message to its one-hop neighbors. After receiving the *point-to-me* message, the neighbor nodes set the storage node as its next hop in its routing table. This mechanism can guarantee that any routing path will eventually lead to the storage node if it can reach the sensor nodes that are one-hop away from the storage node. We define the circle covering these one-hop neighbors to the storage node as the *storage circle*.

The basic scheme provides a simple but effective approach for protecting storage location privacy. However, this scheme may not work well under certain complicated scenarios such as (1) when some sensor nodes are compromised shortly after deployment, (2) when sensor nodes are not deployed at the same time, and (3) when some sensor nodes cannot find their next-hop nodes successfully due to coverage voids or node failures. To address these issues and make the basic scheme to be more secure, flexible and reliable, we will describe several enhancements to the basic scheme in Section 3.3.

3.2.2 Data Deliverability Analysis

We now study the data deliverability from any sensor node to any storage node when the basic scheme is used.

Lemma 1 In the basic randomized storage concealment scheme, the distance of a data message to the target storage node decreases monotonically, and when $d \geq 2r$, where d is the distance between the forwarding sensor node and the target storage node, and r is the communication range, the distance decrement (Δd) at this forwarding hop is at least $d - \sqrt{r'^2 + d^2 - r'd}$, where r' is the minimal one-hop distance.

Proof (sketch) From Figure 3 and the discussion in the previous sections, we know that, as long as λ is selected from $[0, \frac{2\pi}{3}]$, the distance of a data message to the target storage node decreases monotonically, and the distance decrement is

$$\Delta d = d - d_2 = d - \sqrt{d_1^2 + d^2 - 2d_1d \cos \phi}. \quad (4)$$

Since $d \geq 2r$ and $0 \leq \phi \leq \frac{\lambda}{2}$, we have

$$\frac{\partial(\Delta d)}{\partial d_1} \geq 0 \quad \text{and} \quad \frac{\partial(\Delta d)}{\partial \phi} \leq 0. \quad (5)$$

Consequently, since $d_1 \geq r'$ and $\phi \leq \frac{\lambda}{2}$, we have

$$\begin{aligned} \Delta d &\geq d - \sqrt{r'^2 + d^2 - 2r'd \cos \phi} \\ &\geq d - \sqrt{r'^2 + d^2 - 2r'd \cos \frac{\lambda}{2}} \\ &\geq d - \sqrt{r'^2 + d^2 - 2r'd \cos \frac{\pi}{3}} \\ &= d - \sqrt{r'^2 + d^2 - r'd}. \end{aligned}$$

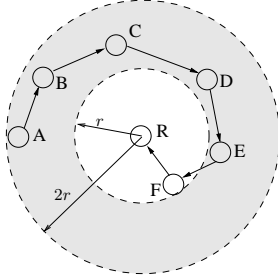


Figure 4: An example data forwarding path when $r < d < 2r$

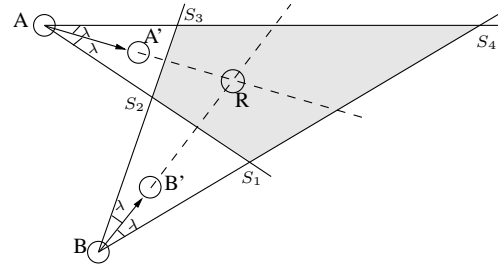


Figure 5: Location inference example with anonymous angle λ

Next, we prove the basic scheme guarantees the data delivery to the storage nodes.

Theorem 1 In the basic randomized storage concealment scheme, each data message from any sensor node will eventually be forwarded to its target storage node.

Proof In the basic scheme, data messages are forwarded by sensor nodes to their next-hop nodes. From the discussion in the previous section, we know that any sensor node's next-hop node has a smaller distance to the storage node than the sensor node itself. Hence, the distance of a data message to its target storage node decreases monotonically along being forwarded. Eventually, the data message will reach the storage circle. Since the storage node is the next hop of any node within the storage circle, the data message will then be forwarded to the storage node.

3.2.3 Communication Overhead Analysis

Based on the property of the basic scheme, we can derive the upper bound of the hop count of the data forwarding path from each sensor node to a storage location (denoted by R) as follows. Suppose that the distance between the sensor node and R is d , the communication range is r , and the minimum one-hop distance is r' . Let's consider the following three cases:

- (i) $d \leq r$: In this case, the sensor node is within the storage circle. So, the hop count is simply one.
- (ii) $r < d < 2r$: In this case, the sensor node is within a donut area around the storage node, which is shown as the shaded region in Figure 4. Let N_2 be the average number of sensor nodes within such donut area. According to our concealment scheme, the distance of a data message to the storage node decreases monotonically along the forwarding path. Therefore, the message will never be forwarded to outside the donut area, or pass the same sensor node within the donut area twice. In other words, once a data message reaches the donut area, it will pass through the area in at most N_2 hops. Once it reaches the storage circle, it will be forwarded to the storage node directly, as illustrated in Figure 4.

(iii) $d \geq 2r$: Since $r \geq r'$, we can denote d as $d = kr'$ with $k \geq 2$ being a real number. From Lemma 1 and Figure 3, we know that

$$\Delta d = d - d_2 \geq d - \sqrt{r'^2 + d^2 - r'd}, \quad (6)$$

and the equality holds when $d = r'$. Consequently, we have

$$\begin{aligned} \Delta d &\geq d - \sqrt{r'^2 + d^2 - r'd} \\ &= r' \left(k - \sqrt{k^2 - k + 1} \right) \\ &\geq (2 - \sqrt{3}) r'. \end{aligned} \quad (7)$$

The equality in (7) holds when $k = 2$, because $(k - \sqrt{k^2 - k + 1})$ is a monotonically-increasing function of k when $k \geq 2$. Combining the maximum hop counts derived for Cases (i) and (ii), we can obtain the upper bound of hop count in this case as $O\left(\frac{d-r}{(2-\sqrt{3})r'}\right) + N_2 + 1 \leq O\left(\frac{d}{(2-\sqrt{3})r'} - \frac{1}{2-\sqrt{3}}\right) + N_2 + 1 = O\left(3.732\frac{d}{r'} - 3.732\right) + N_2 + 1$.

Upper bounds correspond to the worse-case scenarios. In practice, the basic scheme yields comparable performance with that of greedy GPSR schemes, which will be demonstrated in Section 3.2.5.

3.2.4 Security Analysis

In this section, we analyze the effectiveness of the basic scheme in protecting the storage location privacy in the presence of compromised sensor nodes. In the scheme, when a single sensor node is compromised, the adversary can only get the IDs of its next-hop nodes. Such information alone is useless to infer the storage location if the adversary does not know the physical locations of the next-hop nodes. In other words, the adversary can only make use of the information stored at a pair of compromised sensor nodes if they are neighbors along the routes towards the storage nodes.

Without introducing the randomness into selecting the next hops at each sensor node, the information stored at neighboring nodes are highly correlated. Two such pairs will be enough to produce a fairly accurate estimate of the storage location, as discussed in Section 3.1 and illustrated in Figure 1(b), particularly when the sensor nodes are densely-deployed in the sensing field.

In comparison, our scheme introduces the anonymous angle λ , which enlarges an inference line to be a 2λ sector for each pair of compromised neighboring nodes. As shown in Figure 5, A, A' and B, B' are two pairs of compromised neighbors. Even with the direction information about AA' and BB' as well as the knowledge of the anonymous angle λ , the adversary can only infer that the storage node R must be within the polygon area $\overline{S_1 S_2 S_3 S_4}$, which is called the *anonymous area*. The anonymous area may not necessarily be enclosed. The set of sensor nodes within the anonymous area is called the *anonymity set*, and the number of sensor nodes in the set is called the *anonymity set size*.

The anonymity set size varies with the node density, the anonymous area, the number of compromised neighboring node pairs, and the anonymous angle of the adopted concealment scheme. It is an important performance metric to quantify the effectiveness of our scheme, since the adversary has to compromise all the sensor nodes in the anonymity set in order to reveal the storage location. The anonymity set size represents the *absolute cost* for the adversary to launch a successful attack and, in general, a larger anonymity set size implies a better security performance (i.e., a higher privacy level). Another interesting metric is the *anonymity ratio*, which is defined as the ratio of the anonymity set size to the total number of sensor nodes in the network. It reflects the *relative cost* for the adversary to launch a successful attack given a specific network size.

3.2.5 Performance Evaluation

We evaluate the security performance, i.e., the achieved level of storage location privacy, of the basic scheme using a custom simulator. In the simulation setup, we randomly deploy 300, 900 and 2700 sensor nodes into a (600 m × 600 m) square sensing field, and the average communication range is set to 40 m. We randomly pick a node as the storage node and let other sensor nodes construct their next-hop routing tables towards this storage node using our scheme. Then, we randomly compromise a set of neighboring node pairs along the route towards the same storage node. The direction of each node pair is treated as the angular bisector to draw a 2λ sector as shown in Figure 5. The intersection of these 2λ sectors is the anonymous area, and the total number of sensor nodes in this area is the anonymity set size. The anonymity ratio is computed by dividing the anonymity set size with the total number of sensor nodes in the field. We study the impact of the number of compromised pairs on the anonymity set size and the anonymity set ratio with various anonymous angles: $\lambda \in \{\frac{2\pi}{3}, \frac{5\pi}{9}, \frac{\pi}{2}, \frac{\pi}{3}, \frac{\pi}{4}, \frac{\pi}{6}\}$. Simulation results are plotted in Figure 6, and each point in the figure is

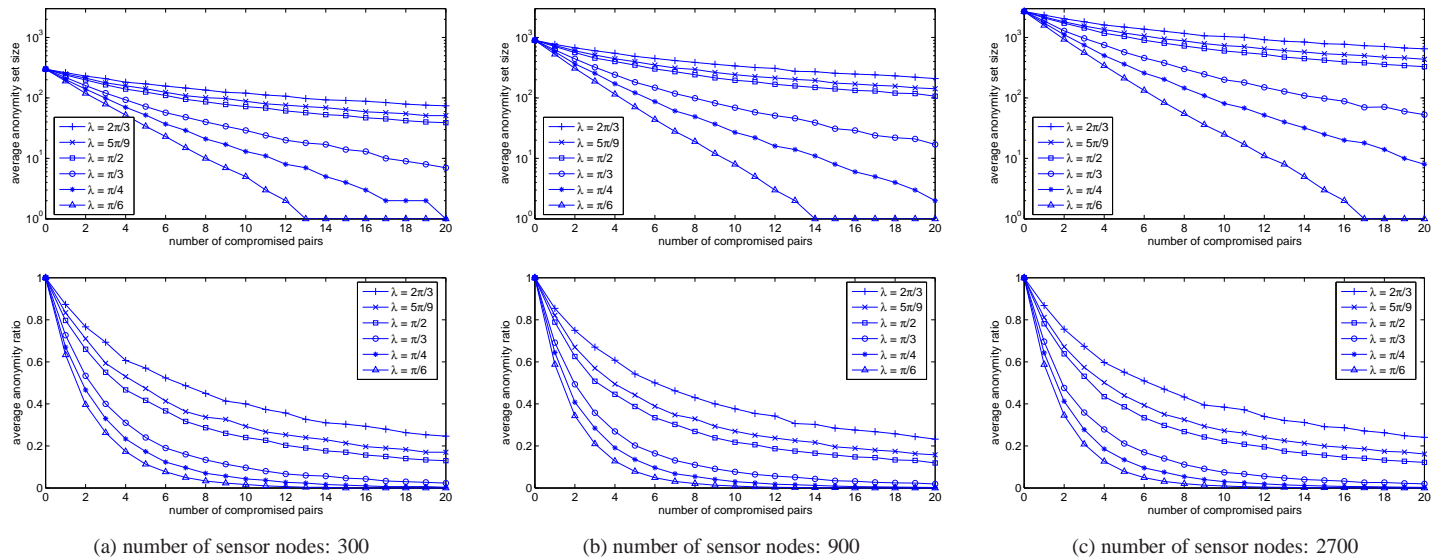


Figure 6: The basic randomized storage concealment scheme is effective in protecting the storage location privacy.

averaged over 500 simulation runs. Hence, results can be viewed as statistical means.

We have the following observation from the figure. Since the anonymous angle corresponds to the degree of randomness introduced into the data forwarding path towards the storage node, with a larger anonymous angle, the anonymity set size is improved and consequently the security performance. The performance gain is more significant when the number of compromised pairs is large. For example, when $\lambda = \frac{2\pi}{3}$, our scheme shows excellent security performance: even when 20 pairs of neighboring nodes have been compromised in the sensor networks of 300, 900, and 2700 sensor nodes, the average anonymity set sizes are 74, 209, and 650, respectively, with the anonymity ratio of $\gtrsim 24\%$.

We also compare the average hop count between a sensor node and a storage node with various anonymous angles, and the results are listed in Table 1. The normalized average hop counts are also shown in the table, which are calculated by normalizing over the average hop counts of the greedy GPSR routing scheme [27]. In general, the length of the data forwarding path with our scheme is comparable with that of the greedy GPSR routing scheme, regardless of the anonymous angle. In particular, when $\lambda = \frac{2\pi}{3}$, the resulting data forwarding path is only stretched by 1.53 times even in a dense network with a very high average node degree of 39. This, combined with the earlier observation of $\lambda = \frac{2\pi}{3}$'s excellent security performance, suggests that $\lambda = \frac{2\pi}{3}$ is a good choice for our concealment scheme.

Table 1: Comparison of AHC (Average Hop Count) and NHAC (Normalized AHC) with different anonymous angle λ

λ		$\frac{2\pi}{3}$	$\frac{5\pi}{9}$	$\frac{\pi}{2}$	$\frac{\pi}{3}$	$\frac{\pi}{4}$	$\frac{\pi}{6}$
300 sensors (avg node degree: 4)	AHC	18.6	18.5	18.5	18.5	18.5	18.4
	NAHC	1.01	1.01	1.01	1.01	1.00	1.00
900 sensors (avg node degree: 12)	AHC	10.8	10.2	9.95	9.41	9.21	8.99
	NAHC	1.31	1.24	1.21	1.14	1.12	1.09
2700 sensors (avg node degree: 38)	AHC	10.6	9.94	9.66	9.02	8.79	8.57
	NAHC	1.53	1.43	1.39	1.30	1.27	1.23

3.3 Enhancements to the Basic Scheme

3.3.1 Enhancement to Achieve a Higher Level of Security

In this section, we describe an enhancement to the basic scheme, which does not require that each sensor node is preloaded with hash table H before deployment. Instead, this enhancement takes the advantage of deployment knowledge and preloads each sensor node with perturbed information of storage locations. Specifically, it assumes that sensor nodes are deployed in groups: [28–30]

- The sensing field is divided into m grids, denoted as $Grid\ 1, \dots, Grid\ m$.
- Sensor nodes to be deployed are also divided into m groups, denoted as $Group\ 1, \dots, Group\ m$.
- Sensor nodes belonging to $Group\ i$ ($i = 1, \dots, m$) are to be deployed into $Grid\ i$.

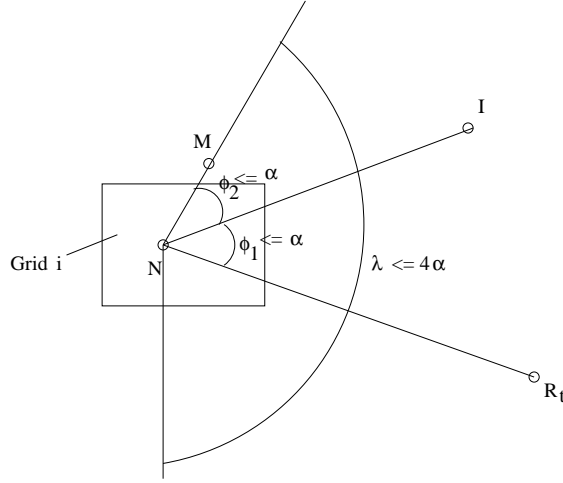


Figure 7: Illustration of the Enhanced Scheme

Let R_t denote the location of storage node for type t . Aided by Figure 7, we detail the process for preloading perturbed information of R_t to nodes in an arbitrary *Grid i*, as follows: Let $\alpha \leq \frac{\pi}{6}$ be a parameter. For any *Grid i* that does not cover R_t , an arbitrary point I is picked such that, for any sensor node N deployed in *Grid i*, $\angle R_t N I \leq \alpha$. Then, the location of point I is preloaded to all nodes in *Grid i* as the perturbed information of R_t . After node N is deployed, it follows the same scheme presented in Section 3.2.1 to discover its neighbors and randomly picks its next-hop node towards node I such that $\angle I N M \leq \alpha$. Then, the location information of I is removed immediately. Since $\angle R_t N I \leq \alpha$, we have $\angle R_t N M \leq \angle R_t N I + \angle I N M \leq 2\alpha$. Further since $\alpha \leq \frac{\pi}{6}$, as shown in Figure 7, we have $\lambda \leq 4\alpha \leq \frac{2\pi}{3}$; hence, message sent by node N can reach R_t according to the message deliverability analysis in Section 3.2.1. For the grid that covers R_t , the location information of R_t is preloaded to all the nodes that are to be deployed in the grid. After deployment, the node closest to R_t becomes the storage node, while other nodes pick their next hop nodes towards R_t using the same scheme presented in Section 3.2.1, and removes R_t immediately.

Compared with the basic scheme, the enhancement does not rely on the assumption that each sensor node will not be compromised shortly after deployment, and hence is more secure. For example, suppose that node N is compromised before it removes the location information of I . Then, based on the captured location information of I , the adversary can only derive the storage location R_t is within a sector that is bisected by line NI and has an angle of 2α , which is much less severe than the case when the preloaded hash table is compromised and consequently the storage location is exposed. In practice, the basic scheme should work fine, as the period required for a sensor node to keep the preloaded information is very short, and therefore the probability of the preloaded information to be compromised is very small.

3.3.2 Enhancement to Allow More Flexible Node Deployment

In this section, we propose another enhancement that allows incremental deployment of sensor nodes.

The basic scheme assumes that all sensor nodes are deployed at the same time. However, in practice, sensor nodes may be deployed in multiple stages in order to extend the lifetime or improve the sensing accuracy of the network. Specifically, after a certain number of sensor nodes have been deployed, new sensor nodes can be added later whenever some sensing or communication voids are detected in the network. To support such multi-stage incremental deployment, the based scheme should be enhanced. The enhanced scheme includes the following steps:

- (1) *Information preloading before deployment*: For each sensor node to be deployed in the very first stage of deployment, it is preloaded with the public key of the network controller (denoted as K_c^+) along with the perturbed information of storage node locations. For each sensor node that will be deployed in a later stage, in addition to K_c^+ and perturbed information of storage node locations, it is also preloaded with $\langle u, TS, K_u^+, K_u^-, \{ID, TS, K_u^+\}_{K_c^-} \rangle$, where u is the ID of the sensor node, TS is the time stamp representing the time when the node is deployed, K_u^+ and K_u^- are the public and private keys for this node, K_c^- is the private key of the network controller, and $\{M\}_K$ represents the result of encrypting M with key K .
- (2) *Operations after deployment*: For the sensor nodes deployed in the very first stage of deployment, the basic scheme is executed to construct the routing table. For each sensor node that will be deployed in a later stage, it immediately broadcasts a request $\langle u, TS, K_u^+, \{ID, TS, K_u^+\}_{K_c^-} \rangle$ to its neighbors. Upon receiving the request, each neighbor first verify the validity of the message using its preloaded K_c^+ and also check if the difference between its current clock and TS is within an acceptable interval (i.e., it is a reasonable delay for a node to be deployed and to broadcast a message). If the message passes the verification, the neighbor sends back its location information which is encrypted with K_u^+ . Having received the replies from its neighbors and decrypted them, the newly deployed sensor node can figure out its own location using GPS or some localization techniques such as triangulation, and then proceed to construct its routing table according to the basic scheme. Note that, each sensor node should remove the preloaded private key K_u^- after it has decrypted the replies from its neighbors.

This enhancement allows sensor nodes deployed in later stages to construct their routing tables. Moreover, the locations of neighbors will not be exposed since the information is encrypted. As reported by Wang et al. [31], when ECC (Elliptic Curve Cryptography) is used, each public key operation can be completed in no more than 3 seconds. The one-hop transmission of short messages between the newly deployed node and its neighbors can also be completed in a few seconds. It is difficult for the adversary to compromise the newly deployed sensor node in such a short time.

3.3.3 Enhancement to Provide More Reliable Information Delivery

We now study how to construct and maintain routing tables in the presence of sensing void or node failures.

Dealing with Sensing Voids Sensing void can simply be dealt with by employing the *right-hand rule* in GPSR [27], which is illustrated in Fig. 8. Node A, having no neighbor within angle λ , looks for neighbors in the counter-clockwise order starting from AR. According to this order, A first finds B, and sets B as its next-hop node. Similarly, B picks C as its next hop, and C picks D as its next hop. After the above procedure, any data message originated from or passing through A can be detoured via B, C, and D to eventually reach the storage node (R).

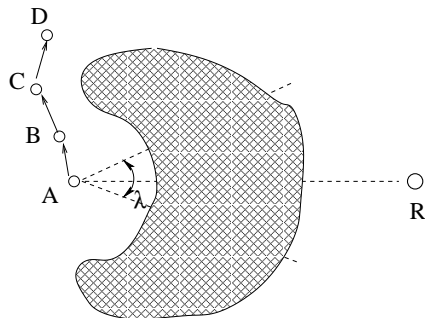


Figure 8: Constructing routing table and forwarding data messages in the presence of sensing void

Keeping Multiple Next-hop Nodes After sensor nodes have constructed their routing tables, some nodes may fail and thus invalidate some links recorded in the tables. To tolerate these failures, each sensor node can pick and keep more than one next-hop node in its *next_hop* field so that, data messages can be forwarded as long as one next-hop node is still alive. In the example shown in Fig. 9, node F has two neighbors G and H recorded in its *next_hop* field, and G records two next-hop nodes I and J. Note that all the next-hop nodes are within angle λ to guarantee data deliverability.

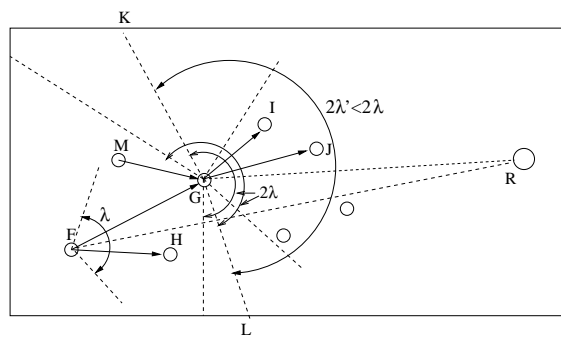


Figure 9: Tolerating failures of next-hop nodes

Backtracking Even with multiple next-hop nodes, all these nodes could possibly fail. To cope with this situation, each sensor node maintains a *previous_hop* field along with the *next_hop* field. The *previous_hop* field keeps the previous-hop nodes which set this node as their next-hop node. In the same example shown in Fig. 9, I and J records G in their *previous_hop* fields, and G sets F as its previous-hop node. With multiple nodes in the *next_hop* and the *previous_hop* fields, the routing procedure is changed to as follows: Normally, each sensor node (e.g., node G in Fig. 9) randomly

picks one node (e.g., node I) from its *next_hop* field to forward its data messages. If all nodes in the *next_hop* field fail or are *flagged* (to be explained later in this paragraph), G resorts to the nodes in its *previous_hop* field to forward the message. Specifically, G forwards its data message to one of its previous-hop node (e.g., F), which then forwards the message to its another next-hop node H. Further, G must let all its previous-hop nodes be aware of the failures of all of its next-hop nodes. For this purpose, G broadcasts a *flag-request* message to all its previous-hop nodes. On receiving the request message, each previous-hop node *flags* G so that G will not be used to forward data messages in the future. If all the nodes in node G's *previous_hop* field fail, node G will request any of its alive neighbor to forward the message for it. Note that this will not cause deadlock since all previous-hop nodes of G have failed and hence the message will not be forwarded back to G.

Impact of The Failure Recovery Enhancements Keeping multiple next-hop nodes may weaken the protection effectiveness of the storage concealment scheme. To study the impact, let us consider the scenario when 900 sensor nodes are deployed into a (600 m×600 m) square sensing field, and $\lambda = \frac{2\pi}{3}$. Every sensor node has 4 neighbors on average covered by its anonymous angle. Assume that every sensor node keeps at most two next-hop nodes for each storage node. In particular, let us consider node G in Fig. 9. Among the 4 neighbors covered by its anonymous angle, we can find out at least 2 nodes (say, I and J) such that $\angle IGJ < \frac{2\pi}{3}/3 = \frac{2\pi}{9}$. If node G keeps both nodes I and J as its next-hop nodes, the *effective anonymous angle* becomes $\lambda' = (2\lambda - \angle IGJ)/2 > \frac{5\pi}{9}$; i.e., if an attacker has compromised node G, I and J, the attacker can conclude that the storage node R must be in the area bounded by KG, GL and the network boundary, where $\angle KGL = 2\lambda'$. According to Fig. 6 (b), when 20 pairs of neighbors have been compromised, the anonymous ratio is $\gtrsim 18\%$, about 25% lower than the anonymous ratio when keeping only one next-hop node. In fact, as the network density increases, the reduction of anonymous ratio becomes even smaller.

The proposed backtracking mechanism may increase hop counts when some sensor nodes fail. Simulations have been conducted to study this impact, and the results are shown in Table 2. As we can see, the increase in hop counts (due to sensor node failures) is generally not significant when the failure ratio is not high. When the failure ratio gets higher, new sensor nodes may be deployed. Our scheme allows the new sensor nodes to participate in protecting storage location privacy; hence, the effectiveness and efficiency of the scheme can be restored.

Table 2: Comparison of the average hop count with different failure ratio of sensor nodes ($\lambda = \frac{2\pi}{3}$)

Failure ratio of sensor nodes	0%	1%	5%	10%
300 sensors (avg node degree: 4)	18.58	19.17	19.41	19.69
900 sensors (avg node degree: 12)	10.82	12.28	13.15	14.33
2700 sensors (avg node degree: 38)	10.63	10.68	12.05	13.79

3.4 Discussion on Other Security Attacks

We discuss some other attacks that can be launched by outside attackers (i.e., attackers have not compromised any sensor nodes and thus do not know any secret keys used in the network) and inside attackers (i.e., compromised sensor nodes that have not been identified yet).

Outside attackers may attack our scheme by analyzing traffic in order to locate storage nodes. To defend against traffic analysis, countermeasures [25, 26] proposed for rate monitoring attacks and time correction attacks can be applied together with our scheme. For example, transmitted messages can be encrypted with the pairwise key shared by the pairs of previous-hop node and next-hop node; messages should not be forwarded immediately by intermediate forwarding nodes but after some random delay; fake messages are generated with some probability to balance the traffic density among different areas. This type of attacks can also be mitigated to a certain degree by our proposed storage migration scheme to be described in next section.

Inside attackers may also attack the randomized storage scheme to locate a storage node with more complicated approaches other than analyzing their routing tables. For example, two non-neighboring inside attackers A and B may collude as follows: If a message sent by A can be received by B later, or vice versa, they can find out that they are on the same path towards some storage node; note that, the information that A and B are on the same path can be used for deriving the storage location, which is similar to the scenario where A and B are a pair of previous-hop and next-hop nodes. To thwart this type of attacks, when a message is relayed by an intermediate node, it may encrypt the message with its key uniquely shared with the storage node (or the network controller). Such encryption prevents B from identifying the message sent by A.

4 Storage Migration

The storage concealment scheme significantly increases the difficulty for the adversary to derive the storage locations. However, if a large number of sensor nodes have been compromised, the adversary will still be able to infer the storage locations based on the partial information kept at compromised sensor nodes. In addition, if some types of data are generated with high rate and transmitted to storage nodes frequently, using a fixed set of storage nodes makes it easy for the adversary to figure out the storage locations via traffic rate monitoring. Moreover, the storage nodes themselves could possibly be compromised or fail for some reasons, even though their locations are not exposed. To address these issues, we introduce a storage migration scheme that is supplemental to the storage concealment scheme. The storage migration scheme forces storage duties to migrate periodically. As illustrated in Figure 10, with storage migration, the anonymity set inferred from previously-compromised sensor nodes become obsolete after the migration, hence further protecting the

storage locations; frequent migration also minimizes the potential damage caused by traffic analysis attacks and storage node compromises.

The storage migration scheme consists of two phases: *migration planning* and *migration execution*.

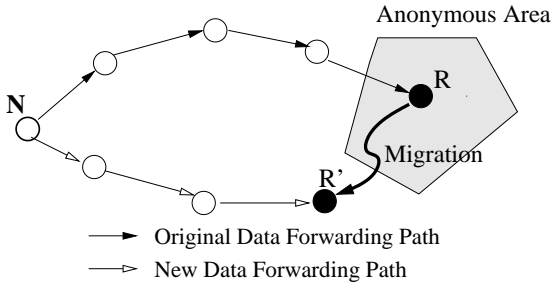


Figure 10: Storage migration: R is the original storage node and R' is the new storage node after migration.

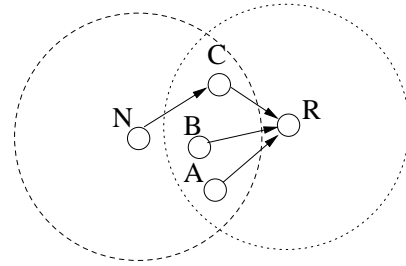


Figure 11: An example migration execution

4.1 Migration Planning

The migration planning phase is performed before sensor nodes are deployed. Aided by Figure 12, we now explain how to plan the migration of the storage duty for an arbitrary data type t (note that the migration of storage duties for other data types can be planned in the similar way). Suppose the original storage node for this data type is node R_0 in *Grid 5*. The next storage node is randomly picked from the nodes that to be deployed in *Grid 9*, denoted as R_1 . Following the similar procedure, the third storage node is determined as R_2 in *Grid 11*, and so on and so forth.

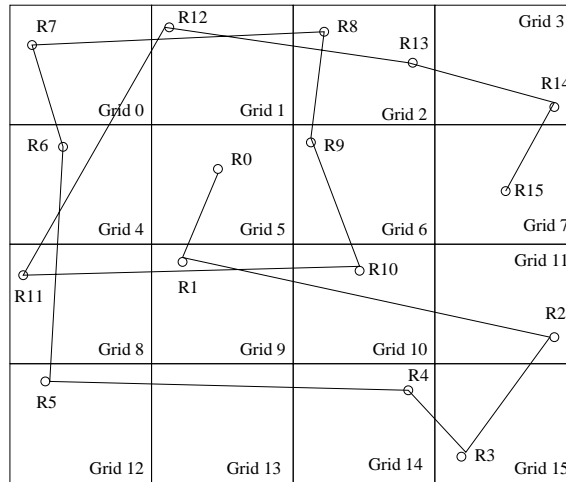


Figure 12: Migration Planning

After the storage nodes have been chosen, secret information should be preloaded such that they can authenticate themselves and meanwhile prevent malicious nodes from impersonating storage nodes. Specifically, the network controller first generates a key chain K_0, K_1, \dots, K_n such that $K_0 = h(K_1) = h^2(K_2) = \dots = h^n(K_n)$. For data type t , the n -th storage node R_n is preloaded with an arbitrary number $C_{t,n}$, which is used as a certificate of its storage duty; the

i -th ($1 \leq i \leq n - 1$) storage node is preloaded with certificate $C_{t,i} = h(C_{t,i+1} | K_{i+1})$. Key K_0 and $C_{t,0} = h(C_{t,1} | K_1)$ are preloaded to each node. Figure 13 shows the relations among the above keys and certificates.

In our storage migration scheme, the storage nodes assigned for period i ($1 \leq i \leq n$) start taking over the storage duties during time window $[T_i, T_i + \Delta T]$; i.e., each of these storage nodes should start taking over the storage duty at an arbitrary time point between T_i and $T_i + \Delta T$. Note that allowing different storage nodes to start their takeover processes at different time points can scatter the traffic caused by the takeover processes, which lowers the effectiveness of traffic analysis attack launched by the adversary. We also assume that ΔT is long enough such that all storage takeover processes for period i can complete by time point $T_i + 2\Delta T$. The time window for the first period of storage migration should be determined before network deployment, while the time windows for the following periods of migration can be determined online. After T_1 is determined, $\delta_1 = h(T_1 | K_1)$ and ΔT are preloaded to every sensor node. The usage of the preloaded information will be explained later when describing the migration execution process.

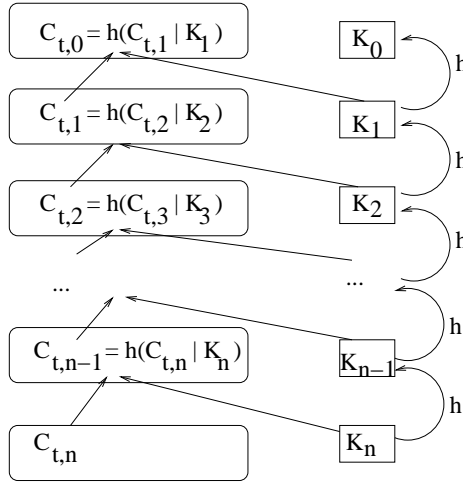


Figure 13: Relations among keys and storage certificates

4.2 Migration Execution

Once the network is deployed, the sensor node closest to the first storage location $H(t)$ of data type t becomes the first storage node for the data type. After all the storage nodes have been established, the network controller broadcasts K_1, T_1 and $\delta_2 = h(K_2 | T_2)$ (here, T_i represents the starting time for the i -th period of migration, and δ_i is used for authenticating T_i). Upon receiving the message, each sensor node can verify the authenticity of K_1 and T_1 by checking if $h(K_1) = K_0$ and $h(K_1 | T_1) = \delta_1$ (recall that K_0 and δ_1 have been preloaded to every node before deployment). Then, δ_2 is stored for later use.

The pre-assigned second storage node for data type t , i.e., R_1 , starts taking over the storage duty at an arbitrary time point during $[T_1, T_1 + \Delta T]$. The takeover process is started by broadcasting a *takeover* message to its neighbors. The

message contains only $C_{t,1}$, hence every receiver can verify the authenticity of the message by checking if $h(C_{t,1}|K_1) = C_{t,0}$; also, the message cannot be marked or changed. In addition, the receiving time of the message should be no later than $T_1 + 2\Delta T$. Upon receiving the first takeover message for a certain data type, each node holds for a predefined time period τ and collects all the takeover messages received during this period. If its current next-hop nodes are the senders of some of these messages, the node keeps its next-hop nodes unchanged and drops the message. Otherwise, the node randomly picks the senders of some of these messages as its new next-hop nodes, and rebroadcasts the takeover message. After time point $T_1 + 2\Delta T$, the network controller will release K_2 , T_2 and $\delta_3 = h(K_3|T_3)$. Then, the above processes will be repeated for the following periods of storage migrations.

Figure 11 shows a simple example of the migration execution. R is the new storage node for a certain data type and it starts broadcasting the takeover message. Nodes A, B, and C receive the message, and after waiting for a time period of τ , all of them choose R as their next-hop neighbor since R is the only node sending the takeover message. Next, A, B, and C rebroadcast the takeover message. Suppose that node N receives all three takeover messages from A, B, and C within the time period of τ . Therefore, it may choose any one of the three nodes as its next-hop neighbor, in this example, node C.

The migration execution protocol guarantees that, for any sensor node, its distance to the storage node is greater than the distance between its next-hop node and the storage node. In particular, considering the example in Figure 11 and letting $d_{X,Y}$ denote the distance between nodes X and Y, we have $d_{C,R} < d_{N,R}$. By derivations similar to (3) in Section 3.2.1, $\angle CNR \leq \frac{\pi}{3}$. Furthermore, due to the randomness in selecting the next-hop node, $\angle CNR$ can be any value between 0 and $\frac{\pi}{3}$. Therefore, the anonymous angle of each sensor node remains $\frac{2\pi}{3}$.

The migration execution protocol allows the network controller to dynamically determine the starting time of migrations. Specifically, the starting time for the i -th period of migration can be decided shortly before the $(i - 1)$ -st period of migration starts. Moreover, as to be discussed in Section 4.4, the protocol can thwart various types of attacks launched by inside or outside attackers.

4.3 Performance Evaluation

4.3.1 Simulation Results on Security Performance

Similar to in Section 3.2.5, we evaluate the security performance of our storage migration scheme using the custom simulator. In the simulation setup, we randomly deploy 900 sensor nodes into a (600 m \times 600 m) square sensing field, and the average communication range is set to 40 m.

First, we vary the number of compromised neighboring pairs and compare the average anonymity set size for the following schemes: the storage concealment and migration schemes with migration distance of 200 m, 100 m, and 20 m,

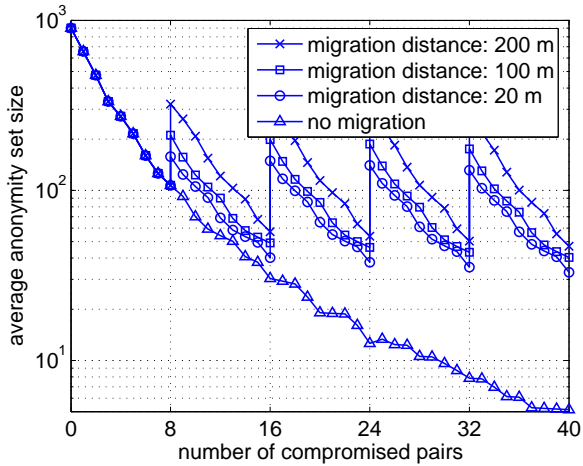


Figure 14: Comparison of security performance (varying migration distances)

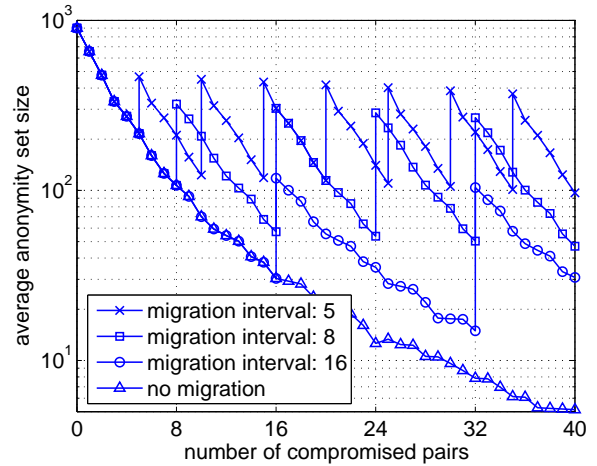


Figure 15: Comparison of security performance (varying migration intervals)

respectively, and the storage concealment scheme (alone without migration). The interval between two consecutive migrations is set to the average time period during which 8 pairs of neighboring nodes may be compromised. Simulation results are plotted in Figure 14. As shown in the figure, after 8 pairs of neighboring nodes are compromised, the anonymity set size drops drastically from 900 to 113. Without migration, the security performance drops continuously, and after 40 neighboring pairs are compromised, the anonymity set size is only about 5, which is considered extremely vulnerable to security attacks. In comparison, with storage migration, the system becomes more resilient to security attacks, which is evidenced by the much larger average anonymity set sizes when 40 pairs of neighboring nodes are compromised: 46.9, 40.3, and 33 for migration distances of 200 m, 100 m, and 20 m, respectively. Moreover, the effectiveness of the storage migration scheme becomes more significant as the migration distance get larger. This is intuitively true since, with a larger migration distance, it is more likely that the previously-compromised neighboring pairs won't contribute in deriving the new anonymous area.

Second, we fix the migration distance to 200 m in our storage migration scheme, and compare the security performance with different migration intervals. Simulation results are plotted in Figure 15. In general, higher migration frequency helps in improving the security performance.

In summary, we can see that, in order to achieve better security performance, the ideal strategy is to migration to far-away storage nodes, and migrate often. On the other hand, frequent migration to far-away locations incurs higher migration cost, which we investigate in the TOSSIM simulator.

4.3.2 Simulation Results with TOSSIM

We use the TOSSIM simulator to study the relation between the migration distance and the migration cost. This is because TOSSIM simulates closely the actual TinyOS [32] network stack and the operation of Mote sensor nodes [33]

and, hence, may produce more meaningful results than our custom simulator. As shown in Figure 16, more sensor nodes are affected and hence more messages are broadcasted when the migration distance increases. So, it is interesting to strike the balance between the security performance and the migration cost for the storage migration scheme, which is one of the topics we plan to investigate in the future.

We also use TOSSIM to study the effect of storage migration on the hop count of the resulting data forwarding path. Due to the limitation of TOSSIM, we simulate a sensor network of 900 sensor nodes randomly deployed in a (600 m × 600 m) square sensing field. Each sensor node is assigned a unique ID from 0 to 899. In our simulation, we assume that there is only one data type in the network, and randomly generate a storage migration plan with node 0 as the original storage node. The locations of node 0 and 10 storage nodes for following periods are shown in Figure 18. Moreover, we set the waiting time period τ to be 2 seconds in the simulation.

The average hop counts of the resulting data forwarding paths in the cases with and without storage migration are compared in Figure 17. Consider the storage node 3 for example. Figure 17 reads that, the average hop count for our storage migration scheme is 9.6 while the average hop count value when node 3 serves as the original storage node is 6.5. In other words, with our storage migration scheme, after three migrations 0→1, 1→2, and 2→3 have been executed, the average hop count of the resulting data forwarding paths is stretched by about 1.5 times. In fact, as shown in the figure, storage migration has little effect on stretching the data forwarding paths, which is a nice desirable side effect of our scheme.

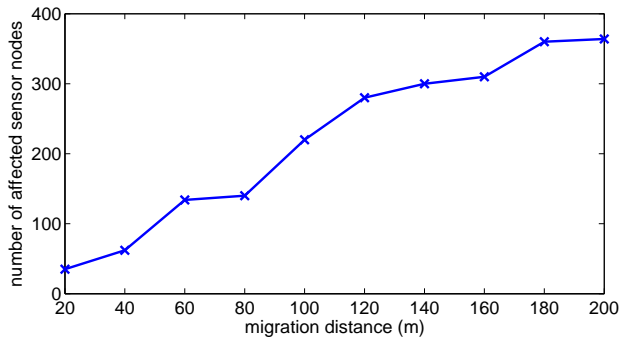


Figure 16: Comparison of per-migration cost

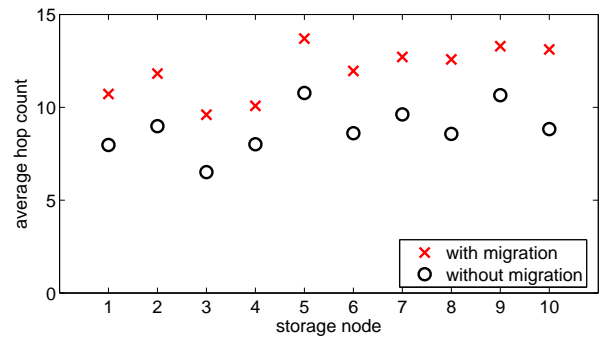


Figure 17: Comparison of hop counts

4.4 Discussion on Other Security Attacks

Outside and inside attackers can both attack our storage migration scheme in the following ways: (1) analyzing the traffic of takeover messages in order to locate storage nodes; (2) impersonating storage nodes to attract and then drop data messages.

For an arbitrary data type t , the certificate for period i (i.e., $C_{t,i}$) can only be derived from certificates for later periods through one-way hashing operations. Specifically, $C_{t,i} = h(C_{t,i+1}|K_{i+1}) = h(h(C_{t,i+2}|K_{i+2})|K_{i+1}) = \dots$. Due to the one-way property of the derivation, an inside attacker holding a certificate for an earlier period cannot figure out any certificate for a later period; that is, the attacker cannot impersonate a storage node for a later period. (3) Suppose an inside attacker was pre-assigned as a storage node for data type t for some period i . The attacker cannot derive the certificate for period $i - 1$ before K_i is released because, the certificate for period $i - 1$ is derived as $C_{t,i-1} = h(C_{t,i}|K_i)$. Also note that K_i is released only after the takeover processes for period $i - 1$ have completed, the attacker cannot impersonate a storage node of period $i - 1$ even after it has derived $C_{t,i-1}$. To generalize, an inside attacker holding a certificate for a later period cannot impersonate a storage node for an earlier period.

Other attacks by inside attackers After an inside attacker receives some takeover message, it may not cooperate in re-broadcasting the message. Specifically, the attacker may drop the message or fabricate the message. Both cases can be tolerated: unless the new storage node is blocked in communication, its takeover message can be propagated by innocent nodes. An inside attacker may choose to cooperate in propagating takeover messages but later drop data messages forwarded to them. This attack can be tolerated by allowing each node to pick more than one next-hop nodes, and alternatively use these next-hop nodes or let multiple next-hop nodes forward redundant data messages.

A special inside attacker is a pre-assigned storage node which is compromised before or after taking the role of storage node. During the period such a compromised node is an effective storage node, all data messages sent to it could be lost, though some data confidentiality protection schemes [34] exist to prevent the adversary from interpreting the data. However, our scheme can mitigate the above damage by forcing storage duties to be migrated periodically to limit the period under attacks.

5 Conclusions and Future Work

In this paper, we proposed a randomized storage concealment and a storage migration scheme to protect the privacy of data storage locations in Data-Centric Storage (DCS) systems. The storage concealment scheme prevents sensor nodes from explicitly keeping the storage node locations; instead, each node only maintains the IDs of its next-hop forwarding nodes on the paths towards the storage nodes. To achieve higher level of location privacy, the storage migration scheme enforces storage duties to be migrated periodically among sensor nodes to thwart accumulative attacks. Extensive analysis and simulations were conducted to verify that the proposed scheme can effectively protect the data storage privacy with modest added overhead.

Although DCS has become a popular strategy for data management in sensor networks, to the best of our knowledge,

there has been very few research on the related security issues. This paper has investigated the important problem of protecting storage location privacy, and proposed a set of lightweight solutions. In our future work, we will take into considerations other security attacks, and further enhance our storage concealment and migration schemes. Particularly, in the near future, we will investigate how to balance the security performance and the migration cost; we will also study how to prevent inside attackers from disrupting the storage migration via wormhole attacks [35].

References

- [1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*, vol. 38, no. 4, March 2002.
- [2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor network," *Proc. of the Hawaii International Conference on System Sciences*, January 2000.
- [3] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication," *MobiCOM '00*, August 2000.
- [4] F. Ye, H. Luo, J. Cheng, S. Lu, and L. Zhang, "A Two-Tier Data Dissemination Model for Large-scale Wireless Sensor Networks," *ACM International Conference on Mobile Computing and Networking (MOBICOM'02)*, pp. 148–159, September 2002.
- [5] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: A Geographic Hash Table for Data-Centric Storage," *ACM International Workshop on Wireless Sensor Networks and Applications*, September 2002.
- [6] A. Ghose, J. Grobklags and J. Chuang, "Resilient data-centric storage in wireless ad-hoc sensor networks," *Proceedings the 4th International Conference on Mobile Data Management (MDM'03)*, pp. 45–62, 2003.
- [7] B. Greenstein, D. Estrin, R. Govindan, S. Ratnasamy and S. Shenker, "DIFS: A Distributed Index for Features in Sensor Networks," *First IEEE International Workshop on Sensor Network Protocols and Applications*, May 2003.
- [8] J. Newsome and D. Song, "GEM: graph eMbedding for routing and data-centric storage in sensor networks without geographic information," *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys)*, pp. 76–88, 2003.
- [9] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin and J. Heidemann, "An Evaluation of Multi-resolution Search and Storage in Resource-constrained Sensor Networks," *ACM Sensys'03*, 2003.
- [10] P. Desnoyers, D. Ganesan, and P. Shenoy, "TSAR: A Two Tier Sensor Storage Architecture Using Interval Skip Graphs," *In Proc. SenSys'05, San Diego, California, USA*, November 2-4 2005.
- [11] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards Sensor Database Systems," *Proceedings of the Second International Conference on Mobile Data Management*, January 2001.
- [12] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin and D. Ganesan, "Building efficient wireless sensor networks with low-level naming," *Proceedings of the Symposium on Operating Systems Principles*, pp. 146–159, October 2001.
- [13] B. Greenstein, S. Ratnasamy, S. Shenker, R. Govindan, and D. Estrin, "DIFS: a distributed index for features in sensor networks," *Ad Hoc Networks*, vol. 1, no. 2-3, pp. 333–349, 2003.
- [14] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 4, no. 2, February 1981.
- [15] M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," *ACM Transactions on Information and System Security*, vol. 1, no. 1, June 1998.
- [16] "US Naval Observatory (USNO) GPS Operations," <http://tycho.usno.navy.mil/gps.html>, April 2001.
- [17] N. Bulusu, J. Heidemann, and D. Estrin, "GPS-less Low Cost Outdoor Location For Very Small Devices," *IEEE Personal Communication, Special Issue on "Smart Space and Environments"*, October 2000.
- [18] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Infocom*, 2004.

- [19] L. Eschenauer and V. Gligor, "A Key-management Scheme for Distributed Sensor Networks," *The 9th ACM Conference on Computer and Communications Security*, pp. 41–47, November 2002.
- [20] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *IEEE Symposium on Research in Security and Privacy*, 2003.
- [21] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," *The 10th ACM Conference on Computer and Communications Security*, 2003.
- [22] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks," *The 10th ACM Conference on Computer and Communications Security*, 2003.
- [23] P. Kamat, Y. Zhang, W. Trappe, and C. Ozturk, "Enhancing Source-Location Privacy in Sensor Network Routing," *The 25th International Conference on Distributed Computing Systems (ICDCS)*, June 2005.
- [24] C. Ozturk, Y. Zhang, and W. Trappe, "Source-Location Privacy in Energy-Constrained Sensor Network Routing," *ACM SASN*, pp. 88–93, 2004.
- [25] J. Deng, R. Han, and S. Mishra, "Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks," *The first IEEE/CerataNet Conference on Security and Privacy in Communication Networks (SecureComm 2005)*, Athens, Greece, pp. 113–124, September 2005.
- [26] J. Deng, R. Han, and S. Mishra, "Intrusion Tolerance and Anti-traffic Analysis Strategies for Wireless Sensor Networks," *IEEE 2004 International Conference on Dependable Systems and Networks (DSN'04)*, Florence, Italy, June 2004.
- [27] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *The Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000)*, Aug. 2000.
- [28] Z. Yu and Y. Guan, "A key pre-distribution scheme using deployment knowledge for wireless sensor networks," *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN 2005)*, April.
- [29] L. Zhou, J. Ni and C. Ravishankar, "Supporting secure communication and data collection in mobile sensor networks," *IEEE Infocom*, 2006.
- [30] W. Du, J. Deng, Y. Han, and P. Varshney, "A Pairwise Key Pre-distribution Schemes for Wireless Sensor Networks," *The 10th ACM Conference on Computer and Communications Security*, 2003.
- [31] H. Wang and Q. Li, "Efficient implementation of public key cryptosystems on mote sensors (short paper)," in *International Conference on Information and Communication Security (ICICS)*, LNCS 4307, Raleigh, NC, Dec. 2006, pp. 519–528.
- [32] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *Proc. of ASPLOS IX*, 2000.
- [33] CROSSBOW, "Wireless sensor networks," http://www.xbow.com/Products/Wireless_Sensor_Networks.htm.
- [34] N. Subramanian, C. Yang and W. Zhang, "Securing Distributed Data Storage and Retrieval in Sensor Networks," *IEEE Conference on Pervasive Computing and Communications (PerCom)*, April 2007.
- [35] Y. Hu, A. Perrig, and D. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks," *IEEE Infocom*, April 2003.