Network Coding based Dependable and Efficient Data Survival in Unattended Wireless Sensor Networks

Wei Ren
School of Computer Science, China University of Geosciences, Wuhan, China
Email: weirencs@cug.edu.cn
Junge Zhao
Department of Information Security, Naval University of Engineering, Wuhan, China
Email: zhaojg36612@163.com
Yi Ren
Information and Communication Technology Department, University of Agder (UIA), Norway
Email: yi.ren@uia.no

***Abstract*— In Unattended Wireless Sensor Networks(UWSNs), sensed data are stored locally for a long term till to collector's retrieval. It is motivated by the scenarios where only historical information or digest data, not real-time data, are of interest. Such paradigm indeed has been attracted more and more interests in research communities recently. Data storage in UWSNs should be dependable to defend random failure or node compromise, as well as the efficiency of communication and storage should be maintained. In this paper, we propose a dependable and efficient data survival scheme to maximize the data survival degree upon data retrieval. Our scheme makes use of computational secret sharing to achieve fault tolerance and compromise resilience, and takes advantages of network coding to further improve communication efficiency. As justified by our extensive analysis, the proposed scheme has the most advantages in terms of robustness and lightweight.**

***Index Terms*— Network Coding, Data Survival, Distributed Storage, Security, Wireless Sensor Networks**

I. INTRODUCTION

Wireless Sensor Networks (WSNs) has been envisioned versatile applications in both civil and military scenarios [1], [2], owing to sensed data that are collected from large amounts of nodes. In most previous researches on WSNs, sensed data are assumed to be collected in a real-time manner. That is, sensed data are forwarded or aggregated to external sinks and stored thereby. The remaining is the other paradigm, called Unattended WSNs (UWSNs) [3], where sensed data are stored locally for a long term or aggregated to certain in-networking data centric nodes.

UWSN paradigm is motivated by scenarios where not real time information, but digest information, are of interest. The digest information includes historical summarization or local results on data processing. For example, what is average temperature during last three months; what is the highest and lowest humidity degree during last 24 hours; or more specifically, what is average content of a chemical element in soil during last half year [4]. The digest information is provided on demand upon user's retrieval, avoiding frequent data transmission to sinks. Thus it remarkably saves the communication energy. Moreover, there exist some scenarios that robust real-time sinks may not be convenient to deploy, e.g. in the sea, volcano, or located far away from the collector. The data have to be stored locally or designated nodes in the network and wait for further collection.

Currently, UWSN paradigm attracts more and more interests in research communities [3]–[6]. For example, in "Unattended WSNs" [3], data collection is not performed in (or near) real time and data should survive for a long time enough to be collected. "In-Situ Data Storage WSNs" [4] store the sensor readings at the generating sensor (called In-Situ) node ; "Storage-Centric WSNs" [5] store historical data for the applications that need to mine sensor logs to analyze historical trends; In "Asynchronous WSNs" [6], sensor nodes do not transmit the data to authorized devices in real-time, but store the data itself; In "Data Centric WSNs" [7], relevant data are stored by name at nodes within the network, and all data with the same general name will be stored at the same node. [3]–[10]. In-networking storage is now possible and available with the performance improvement of the sensor node, in particular, storage capabilities [11]–[13] and computation capabilities [14]–[16].

Generally, users or mobile collectors visit UWSNs periodically to collect data. Because the nodes always expose outside for a long term, they may malfunction due to some threats, e.g., physical failure such as melting, corroding or getting smashed, and more sophisticated, mobile adversary attacks [17]. Some data may be lost, erased or modified before the arrival of the mobile collectors, which significantly affects functionalities of UWSNs. Therefore, the critical issue for UWSNs is: how to maximize data survivability. Or, how to maximize the portion of correct data upon collection of data.

This work is supported in part by the Science and Research Start-Up Project for the Recruit Talent of China University of Geosciences under Grant No.20090113.

doi:10.4304/jcm.4.11.894-901

Recent research in [3] starts to address this problem, but thorough exploration are still remained. Intuitively, certain methods that can grantee data recovery via redundancy or source coding can be applied, but efficiency may not be straightforward to attain in the context of WSN's resource constraints. The tailored design is required by accommodating both robustness and efficiency.

Network coding can improve the network efficiency such as maximize network throughput, reduce network delay and enhance energy efficiency, which have various applications in both traditional networks and emerging networks. In this paper, we explore how to apply computational secret sharing to guarantee robustness, bounded dispersing, and network coding to improve both robustness and efficiency.

Objective of the paper: The goal of this paper is to investigate techniques that can maximize data survival in UWSNs in presence of random failure and node compromise, and yet maintain high energy consuming efficiency as well.

Contributions: (1) We investigated possible techniques for data survival and analyzed their performances in terms of proposed metrics - data survival degree. (2) We proposed a computational secret sharing based scheme to maximize communication and storage efficiency and data survival degree. (3) We proposed a network coding based enhanced scheme to further improve the communication power consuming efficiency.

Organization: The remainder of the paper is organized as follows: The network assumption, adversary model and design goal are presented in Section II. Section III reviews the preliminaries and outline the notations. Section IV presents the proposed schemes in a comparatively progressive manner. In Section V, we analyze the performance and security of the proposed schemes. Section VI briefly reviews the related work. Finally, Section VII concludes the paper.

II. Problem Formulation

A. Network Assumptions

Suppose N sensor nodes exist in WSNs, denoted as a graph $G(V, E)$. The node set is $V = \{v_1, v_2, ..., v_n\}$ and edge set is $E = \{e_1, e_2, ..., e_m\}$. The node is $v_i (1 \leq i \leq N)$. The node v_i has neighbors that composes a set Set_i.

The sensor nodes have enough memory to store data, e.g. the Gigabytes memory for sensor node is already available [18]. The communication mode between nodes is peer-to-peer message or local broadcasting. The local broadcasting message at node v_i can be received by nodes in Set_i. The pairwise key between nodes is already deployed in the networks.

Mobile collectors collect all the data stored in sensors occasionally or periodically. Once data are collected, if needed the sensor nodes can be recovered back to its original status that no mobile adversary exists, e.g. by clear of memory, reboot or replace old with a new one. Thus mobile adversary's attcks are not accumulative and the discussion is focused within one access interval, as assumed in [3].

B. Attack Model

Physical Failure ($Attack^{pf}$). Sensor nodes are vulnerable to the physical corruption such as smash, melt, or corrode, intentionally by attackers or not. Such attack leads to the completely loss of the stored data in the node.

Mobile Adversary ($Attack^{ma}$). We also assume more powerful adversary such as mobile adversary [17] and node capture [19] occurs. The mobile adversary may compromise the sensor nodes by exploiting the pitfalls in the codes of the sensor, and subvert to erase or modify some data like a parasite. They may also spread to other nodes like a worm virus. We observe that the pattern of the mobile adversary has *Space Correlation* property. We then model such property to facilitate the design of defense schemes.

Definition 1: Space Correlation of Mobile Adversary If the sensor node in location (x, y) is detected to be compromised due to mobile adversary, the nodes in $(x + \Delta x, y + \Delta y)$ are also very likely to be attacked.
As justified by the empirical observation, the nodes suffered by mobile adversary are always near to each other.

Randomly fault of data in the sensor node is also included as a special case in mobile adversary. Randomly fault may come from transmission errors, signal interference, channel competition loss, or partial power depletion. Different from physical failure, it results in the partial data modification or loss. We thus look it as a mobile adversary with zero mobility speed.

We focus on data erasion induced by mobile adversary because the detection of data modification can be easily achieved by exploiting MAC (Message Authentication Code). Since mobile adversary always desires to avoid being detected, data erasion portion normally is small to avoid the intrusion detection alert [20]. We thus assume the mobile adversary can only erase data with the limit portion P_{ma} within one access interval. Otherwise, more rate of erasions will trigger the alter such as intrusion detection.

C. Design Goal and Evaluation Metrics

We first define the metrics to measure data survival level.

Definition 2: Data Survival Degree(DSD) DSD is a ratio - the valid data that mobile collector obtains over total data that sensor nodes have sensed since the last visit of mobile collectors.
The goal of the design is to maximize *DSD* in presence of $Attack^{pf}$ and $Attack^{ma}$, denoted by DSD^{pf} and DSD^{ma}, respectively. The *DSD* of the defense scheme S corresponding to such attacks are DSD_S^{pf} and DSD_S^{ma}. Moreover, the cost in terms of power consumption should be manageable. The power consumption has three parts: communication, storage and computation. In such three ones, we focus on communication cost because communication costs much more energy than storage. For

example, transmitting data over radio channel consumes 200 times more energy than storing the same amount of data locally on a sensor node [21]. Radio reception costs 500 times more energy than reading the same amount of data from local storage [18]. Computation cost depends on the involving calculation. If only simple arithmetic operations such as addition or multiplication, it may have relatively low cost due to the powerful processor in the new generation sensor node [14], [22]. Therefore, we focus on the communication cost.

III. Proposed Schemes

A. Basic Scheme I: Data-Moving Scheme

Firstly, we consider Do-Nothing scheme as a base line, in which nothing is employed to defend attacks. Obsequiously, physical failure will destroy certain data if nothing is employ in defense. More specially, $DSD^{pf}_{Do-Nothing}$ is:

$$DSD^{pf}_{Do-Nothing} = 1 - \frac{N_{pf}}{N},$$

where $DSD^{pf}_{Do-Nothing}$ is DSD of Do-Nothing scheme in presence of physical failure; N_{pf} is number of nodes destroyed by $Attack^{pf}$; N is number of total nodes.

In Data-Moving scheme, when a node senses new data, it randomly selects one node in its neighbors and moves data to it. Data-Moving has two basic strategies [3]. One is MOVE-ONCE, in which the sensed data is moved to one neighbor and then stop moving; The other is KEEP-MOVING, in which the sensed data is moved to one neighbor and keep on moving to such neighbor's neighbor. In fact, KEEP-MOVING scheme causes large communication overhead and is difficult to deployed in WSNs.

Also, we observe that Data-Moving scheme indeed gains no more advantage than Do-Nothing scheme.

Proposition Data-Moving scheme has the same DSD^{pf} with the Do-Nothing in presence of $Attack^{pf}$.

Proof: (sketch) Straightforward. The destroyed data volume by $Attack^{pf}$ is the same as in the scheme MOVE-ONCE or KEEP-MOVING. Therefore, the DSD^{pf} is same. ■

B. Basic Scheme II: Data-Replicating Scheme

As we observe that trivial Data-Moving scheme cannot improve DSD^{pf}, normally data redundancy method can be relied to improve data dependability. We thus propose Data-Replicating scheme, in which sensed data are replicated and sent to neighbors.

Proposition Data-Replicating scheme has higher DSD^{pf} than Do-Nothing/Data-Moving scheme in presence of $Attack^{pf}$.

Proof: Think about k-Data-Replicating scheme, in which sensing data is replicated to k copies and distributed to neighbors. In presence of $Attack_{pf}$, if $k \leq N_{pf}$, then

$$DSD^{pf}_{k-Data-Replicating} = 1 - \frac{N_{pf}}{k * N}.$$

If $k > N_{pf}$, then $DSD^{pf}_{k-Data-Replicating} = 1$. In both cases, DSD^{pf} is improved. ■

However, naive Data-Replicating scheme suffers from higher data loss due to *Space Correlation* of $Attack^{ma}$. The replicated data is subject to be erased by mobile adversary in corresponding areas. Because the mobile adversary always spreads in related areas, we further conclude that if replicated data can move to other randomly selected nodes, ratio of erased data over all data will become lower. To justify our conclusion, we firstly define data location entropy.

Definition 3: : *Data Location Entropy (DLE).* Let the $H(d, x)$ be the entropy of data d's location entropy.

$$H(d, x) = -P(d, x) log P(d, x),$$

where $P(d, x)$ is probability that data d is located at the node x. When $P(d, x)$ is $1/N$ (N is the number of nodes), $H(d, x)$ achieves the maximum value.

Proposition : The DSD^{ma} is higher if and only if average DLE is larger in presence of $Attack^{ma}$.

Proof: (sketch) When DLE is larger, for a given data d, its location is more uncertain. Thus, it has lower probability to be erased by $Attack^{ma}$. ■

Therefore, the proposed scheme should combine both Data-Replicating to defend physical failure and Data-Moving to defend mobile adversary. However, trivially moving data in the network has no energy efficiency. Thus, to save the energy, data should be moved for limited hops. This limited value can make DLE large enough. We propose that such value equals the average radius of WSNs. Average radius is the average hop distance between two nodes in WSNs, so moving more than such value waste energy without gain of DLE.

Based on such observation, we propose a (α, β)-bounded Data-Replicating scheme. That is, the data is replicated to α copies and each replica moves to randomly selected $\beta \in [1, 2R]$ hops in randomly selected direction. R is the average radius of WSNs in terms of hops.

Proposition : The DSD^{ma} is higher if and only if average DLE is larger in presence of $Attack^{ma}$.

Proof: As β is randomly chosen in $[1, 2R]$, the average moving hop counts are R. The probability of data location reaches to an average value $1/N$, so that the DLE approaches to an maximal value and energy consuming remains efficient. Meanwhile, data is replicated to α ones, the DSD of $(\alpha - \beta) Data - Replicating$ scheme in presence of $Attack^{ma}$ is:

$$DSD^{ma}_{(\alpha-\beta)Data-Replicating} = 1 - P^{\alpha}_{ma}.$$

■

C. Advanced Scheme - Data-Dispersing

We observe that Data-Replicating scheme can be further improved with respect to dependability in terms of compromise resilience. For example, if one replica is revealed by a compromised node, the original data will be exposed. To further improve dependability, we consider to generate and distribute data shares, instead of replica, by using straightforward secret sharing method. However, such way suffers from high storage and communication overheads because the share size is the same as the original data size. We thus propose a computational secret sharing based scheme - Data-Dispersing, which has much smaller share size than naive secret sharing scheme.

Basic Setting:

After sensor node v_i senses data $data$, it follows the listed steps to protect the data integrity and data confidentiality:

(1) Generate a random symmetric key RSK;

(2) Encrypt $data$ using key RSK. The encrypted data is $DATA = Encrypt(data, RSK)$, thus data confidentiality is protected.

Step I: Share Generation:

(1) v_i employs a (m, n) secret sharing scheme to encode RSK into n shares, denoted as $RSK_1, ..., RSK_n$. Any m in such n shares can reconstruct RSK, but any no more than $m-1$ in such n shares reveal zero information on RSK (called information theoretical security).

(2) v_i employs a (m, n) error correcting coding scheme to encode $DATA$ into n shares, denoted as $DT_1, ..., DT_n$; Any m in such n shares can reconstruct $DATA$.

Step II: Share Distribution:

v_i randomly selects n neighbors in Set_i (e.g. v_j is one of them), and distributes one randomly selected distinct share DT_t and $RSK_t, t \in [1, n], t \in Z$ to v_j by using Pairwise Symmetric Key between v_i and collectors $K_{i,c}$ to encrypt the packet. That is, the dispersing n shares are

$$S_t = \{DT_t \| RSK_t\}_{K_{i,c}},$$

where $t = 1, 2, ..., n$; S_t is a dispersing share; v_i is sender's id; DT_t is a piece of encoded $DATA$; RSK_t is a piece of encoded RSK.

Step III: β-Bounded Moving:

To defend mobile adversary, move shares to β-bounded hops so that DLE becomes large enough yet avoid unnecessarily moving to save communication energy. Thus, each share in step II is dispersed to n selected neighbors in a manner as follows:

$$v_i \rightarrow v_j : \{v_i \| TS \| t \| S_t \| CNT\},$$

where $t \in [1, n], t \in Z$; TS is timestamp; t acts as a sequence; $CNT = \beta$; $v_j \in Set_i$. Here $\{v_i \| TS \| t\} = UID$ is a triplet that can uniquely identify a share.

We describe basic functions when v_j receives $\{UID \| S_t \| CNT\}, t \in [1, n], t \in Z$.

Algorithm 1 Receiving function upon receipt of $\{UID \| S_t \| CNT\}$ at v_k

```
if CNT == 0 then
  save {UID||S_t} locally
else
  CNT ⇐ CNT − 1
  randomly selected v_l ∈ Set_k
  send {UID||S_t||CNT} to v_l
end if
```

Step IV: Data Reconstruction:

The collectors collect and decrypt out m shares in nodes and reconstructs RSK using secret sharing and $DATA$ using error correcting coding method. Finally, decrypts $DATA$ using RSK and $data$ is out.

Example:

We give a trivial example to explain the procedure of data-dispersing. Suppose node v_i wants to distribute sensed data $data$, which is 12 bytes. It generates a random symmetric key RSK with 2 bytes encrypt $data_i$ into $DATA = Encrypt(data, RSK)$. v_i transforms $DATA$ into 8 pieces using (6,8) RS coding. At first, divides $DATA$ into 6 pieces: $\{D_1 \| ... \| D_6\}$. Each piece thus has 2 bytes. Let $Z_q = Z_{65537}$. Let $DT_i = D_1 + D_2 i + ... + D_6 i^5$. Calculate DT_i using $i = 1, ..., 8$. The coding results for $DATA$ are DT_i $(1 \leq i \leq 8)$. Each 6 in 8 pieces can recover $DATA$ by resolving the 6 equations. RSK is transformed into 8 pieces by making use of $(6, 8)$ secrete sharing scheme. That is, select random polynomial $a(x)$ with degree 5 and constant term RSK. The shares for RSK are RSK_i, $i = 1, ..., 8$. Share RSK_i is generated by following equation: $RSK_i = RSK + a_1 i + a_2 i^2 + ... + a_5 i^5, i \in [1, 8]$. RSK can be reconstructed by 6 in 8 pieces. Here all operations are in Z_q. The $DATA$ thus can be obtained by $data = Decrypt(DATA, RSK)$. Note that, if naive secret sharing scheme is used to transform DATA, the length of share will be 12 bytes. But, the share length is shortened to $2 + 2 = 4$ bytes by making use of computational secret sharing. That is the reason why we choose computational secret sharing rather than naive secret sharing scheme.

Parameters Selection:

Select n neighbors to distribute shares. To reconstruct data, only m shares are required. Let $\lfloor \gamma * n \rfloor = m$, where γ is a dependability parameter (e.g., $0.15 \sim 0.4$). When γ is larger, more percentage of shares is required for data reconstruction.

D. Enhanced Scheme: Network Coding Based Scheme (NCBS)

The efficiency of (α, β)-Data-Dispersing can be further improved by network coding methods. The major enhancement of is at Step III: β-bounded moving. To save communication overhead, some forwarding nodes encode received shares using linear network coding, instead of simply forwarding them without any changes. It induces some computation energy, whereas communication energy reduces. As the communication consumes more energy than computation, overall energy efficiency is improved.

1) Basic Setting: Recall that network is $G = (V, E)$. Source nodes are $\mathcal{S} = \{v_1, v_2, ..., v_N\} \subseteq V$. Sink nodes are $\mathcal{D} = \{u_1, u_2, ..., u_K\} \subseteq V$. We focus on the operations on the forwarding nodes, say, $f_a \in \mathcal{F}$. Forwarding nodes are $\mathcal{F} = \{f_1, f_2, ..., f_L\} \subseteq V$. The input packets (packets on the inputting edge) of f_a are $X_i, i \in [1, p]$; the output packets of f_a are $Y_j, j \in [1, q]$. Without loss of generality, we concentrate on one output edge, say Y_a. We have:

$$Y_a = \sum_{i=1}^{p} \alpha_i X_i$$

where $(\alpha_1, ..., \alpha_p)$ is encoding vector.

Encoding vector is also used by sinks to recover the source message. They operate in same field with the aforementioned computational secret sharing scheme for simplicity of implementation. We assume each node has already pre-deployed encoding vector. Moreover, each node also have functions to randomly generate encoding vector.

2) Random Linear Network Coding Algorithm: We induce certain linear network coding nodes in the networks via coding the forwarding packets with probability p_{nc}. In this scheme, each node has pre-deployed an encoding vector. Suppose v receives p Ss, denoted as $\{UID_1 \| S_1 \| CNT_1\}, ..., \{UID_p \| S_p \| CNT_p\}$, from others. v encodes p into one packet with the probability p_{nc} and sends it out. We call v Network Coding Node if it indeed encodes the packets. The basic coding method is described in algorithm 2.

For simplicity of the description, we omit packet parsing procedure in algorithm 2, in which the packet type - PT is a signal to distinguish whether the packet is network coded packet or not. Y_i is the encoding results. $AVGCNT$ performs for the optimization of forwarding times. v_i works as the identity for encoding vector. $UID_1, UID_2, ..., UID_p$ are the encoded shares' unique IDs. Note that, each packet is encoded only once. Moreover, $\alpha_{i_t} (t = 1, 2, ..., p)$ compose encoding vector α_i at node i. $Average()$ function returns average of a array.

Proposition : The decoder can finally decode out $S_i, i \in [1, p]$ when given total p numbers of Ss and Ys.

Proof:

Algorithm 2 Encoding function at a network coding node v_i upon receipt of p Ss $(S_1, S_2, ..., S_p)$ whose CNTs are non-zero, such p packets are encoded into one packet.

```
dim ⇐ 0
AVGCNT ⇐ 0
i ⇐ 1
PT ⇐ 1
while (i ≥ 1) do
  if (CNT_i == 0) then
    Save S_i locally
    Continue
  end if
  Dim ⇐ Dim + 1
  Save CNT_i in Array[Dim]
  if (Dim==p) then
    Generate a random number r ∈ [0, 1]
    if (r > p_nc) then
      Forward p packets to others
      Exit
    else
      AVGCNT ⇐ Avgerage(Array[Dim])
      Y = Σ_{t=1}^{p} α_{i_t} S_t
      Send out {PT‖Y_i‖AVGCNT‖v_i‖UID_1...‖UID_p}
      Exit
    end if
  end if
end while
```

Suppose collectors collect p number of data that has the same UID set, not matter encoded or not. Without loss of generality, we denote them as $Y_1, S_2, ..., S_{p-1}, Y_j$. We have:

$$\begin{pmatrix} \alpha_{1_1} & \alpha_{1_2} & ... & ... & \alpha_{1_p} \\ 0 & 1 & 0 & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \alpha_{i_1} & \alpha_{i_2} & ... & ... & \alpha_{i_p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & ... & ... & 1 & 0 \\ \alpha_{j_1} & \alpha_{j_2} & ... & ... & \alpha_{j_p} \end{pmatrix} \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_i \\ \vdots \\ S_{p-1} \\ S_p \end{pmatrix} = \begin{pmatrix} Y_1 \\ S_2 \\ \vdots \\ Y_i \\ \vdots \\ S_{p-1} \\ Y_j \end{pmatrix} \quad (1)$$

Denote above encoding matrix as EV. We have $(S_1, S_2, ..., S_i, ..., S_{p-1}, S_p)^T = (Y_1, S_2, ..., Y_i, ..., S_{p-1}, Y_j)^T EV^{-1}$, where M^T means transpositive matrix of M and M^{-1} is inversive matrix of M. Packets are encoded at rate of p_{nc} for mitigating computational cost. p_{nc} is a tuning parameter.

■

3) Enhancement via Multiple Encoding Vectors: In the previous scheme each node uses the encoding vectors with fixed dimension to encode forwarding packets. Thus, it may have to defer for enough packets to encode (until the incoming packets are the same as the dimension of vector, as denoted in $|ln(T)|$ or p). To further improve the efficiency and flexibility of our scheme, we deploy a bunch of encoding vectors in each node. That group of vectors have various dimension ranged from 1 to p, so that encoding node can select corresponding vector

ν in accordance with the number of incoming packets upon encoding. (Of course, the vectors are also identified by some simple index, such as $\{nodeid \| vectorid\}$, e.g., $\{v_i \| vid\}$ for facilitating decoding.)

Proposition : The decoder can finally decode out $S_i, i \in [1, p]$ when given total p numbers of Ss and Ys.

Proof:

Similar to Proposition 4.5, we have encoding equation as follows:

$$\begin{pmatrix} \nu_1 & 0 & ... & 0 \\ 0 & 1 & 0 & ... \\ \vdots & \vdots & \vdots & \vdots \\ 0 & ... & \nu_i & ... \\ \vdots & \vdots & \vdots & \vdots \\ ... & 0 & 1 & 0 \\ 0 & ... & 0 & \nu_p \end{pmatrix} \begin{pmatrix} S_1 \\ S_2 \\ \vdots \\ S_i \\ \vdots \\ S_{p-1} \\ S_p \end{pmatrix} = \begin{pmatrix} Y_1 \\ S_2 \\ \vdots \\ Y_i \\ \vdots \\ S_{p-1} \\ Y_j \end{pmatrix} \quad (2)$$

Note that, ν_i is a selected vector with the same dimensions as the number of incoming packets upon encoding. The omitted items in the first matrix (encoding matrix) are all zeros. The decoding method is the same. ■

Implementation Consideration:

RS codes, secret sharing and network coding operate in the same Galois Field. The implementation code thus can share the same library. RS can be implemented based on Cauchy matrices or XOR-based RS codes [23], [24] and such implementation is faster for larger values of order m than the implementation based on Vandermonde matrices. Since in our WSNs the m is upper bound the largest neighbor, it is not too large normally. Also, fast Galois Field library is already available for arithmetic in $GF(2^8)$, $GF(2^{16})$ and $GF(2^{32})$, released by Plank and Kevin M. Greenan et al. [25], [26].

IV. Analysis

A. Performance Analysis

We summarize comparisons between Do-Nothing (DN), Data-Moving (DM), Data-Replicating (DR) and Data-Dispersing(DD) in Table I. Table II shows quantitative performance analysis in terms of storage, communication and computation overhead.

TABLE I.
Comparisons between Data Survival Schemes (Physical Failure (PF), Mobile Adversary (MA), Dependability (DEP), Overhead (OH))

Scheme	PF	MA	DEP	OH
DN	no	no	no	no
DN⇒DM	same	improved	improved	increased
DM⇒DR	improved	same	improved	increased
DR⇒DD	same	same	improved	decreased

Theorem 1: : The network coding based scheme (NCBS) can improve the communication efficiency of Data-Dispersing by roughly $\frac{1*p_{nc}}{p}$

Proof: Straightforward. ■

TABLE II.
Quantitative Analysis between Data Survival Schemes

Scheme	Storage Overhead	Communication Overhead	Computation Overhead
DM	$\|DATA_i\|$	$\propto$	-
DR	$\alpha\|DATA_i\|$	$\beta\|DATA_i\|$	-
DD	$\approx n\frac{\|DATA_i\|}{m}$	$< \beta * n\frac{\|DATA_i\|}{m}$	higher

B. Security Analysis

Theorem 2: In case of physical failure and mobile adversary, the data can be reconstructed upon user's request with the probability $\frac{(n(1-p_m))!(n-m)!}{n!(n(1-p_m)-m)!}$, assuming $n(1-p_m) \geq m$ and $p_m = N_f/N + p_{ma}$.

Proof: The data can be reconstructed if the data loss is less than m. The data loss comes from physical failure and/or mobile adversary, so the percentage of failed nodes is $p_m = N_f/N + p_{ma}$. The successful access are $C^m_{n(1-p_m)}$. The total accessible cases are C^m_n. Therefore, the probability that mobile collectors can successfully access data upon the request is:

$$\frac{C^m_{n(1-p_m)}}{C^m_n} = \frac{(n(1-p_m))!(n-m)!}{n!(n(1-p_m)-m)!}.$$

■

Figure 1 depicts the analysis results for some (m, n) parameters.

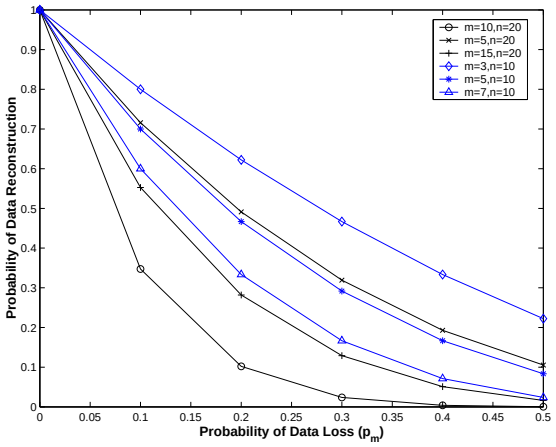

Figure 1. Probability of data reconstruction in presence of physical failure and mobile adversary

Moreover, RSK is generated by Pseudo-random Generation Function. If the length of RSK is l. The secrecy of RSK is $\Omega(2^l)$. RSK is used for only one time, so even if the key is exposed or nodes are compromised by attackers, it can only be used for attacking one data.

V. Related Work

R. D. Pietro et al. [3] pioneer the work of the data survival in sensor networks. They propose a scheme similar to Keep-Moving. They assumed power in sensor network has no constraints, which is unrealistic in the real-world applications. S. Chessa, et al. [27], [28]

propose to employ RRNS (Redundant Residue Number System) for the dependable and secure storage in ad-hoc networks. The residue numbers need to be pre-distributed and keep secret, and once one node is compromised, all residue number will be exposed. Other works such as [29], [30] discuss the secure data storage or aggregation in WSNs, but they not concentrate on data survival.

T. Ho et al. [31] propose how to use distributed randomized network coding to provide Byzantine modification detection without the use of cryptographic functions. S. Jaggi et al. [32] propose algorithms that are information-theoretically secure and rate-optimal for different adversarial strengths of the adversary.

J. Tan et al. [33] investigate the network cost and network security jointly. They consider the situation where a set of message is to be multicasted across the network, and a known subset of these messages is of interest to a wiretapping adversary. They try to find a network coding scheme that has both a low network cost and a low probability of the wiretapper being able to retrieve all the messages of interest.

L. Lima et al. [34] consider the network in which all nodes comply with the communication protocols yet are assumed to be potential eavesdroppers ("nice but curious"). They develop a natural algebraic security criterion and prove several of its key properties. Zhen Yu et al. [35] proposed a homomorphic signature based scheme against the pollution attack, which is more efficient than existing ones.

VI. Conclusion

In this paper, we firstly reviewed the weakness of previous proposed schemes and pointed out the major concerns for data survival via formal proof. We then proposed a computational secret sharing based scheme - Data-Dispersing - to improve dependability and efficiency of data survival in unattended wireless sensor networks. This technique outperforms naive secret sharing scheme in that the share size is smaller but the security is still maintained. We next proposed a network coding based scheme to further enhance the communication efficiency. We proofed data replication can improve the security against physical failure. We also proofed hop bounded data moving can improve security against mobile adversary yet maintain power consuming efficiency. As justified by our extensive analysis, Data-Dispersing scheme has the most advantages in terms of robustness and efficiency for maximizing data survival in UWSNs. Meanwhile, network coding based scheme can further improve communication efficiency as an enhancement of Data-Dispersing scheme. Our scheme includes tailored design of source coding and network coding for UWSNs, which can be implemented via same finite field library.

Acknowledgment

This work is supported in part by the Science and Research Start-Up Project for the Recruit Talent of China University of Geosciences under Grant No.20090113, and was done in part in Department of Electrical and Computer Engineering, Illinois Institute of Technology, USA.

References

[1] T. Wark, C. Crossman, W. Hu, P. Corke, P. Sikka, Y. Guo, P. Valencia, C. Lee, J. O'Grady, and J. Henshal, "The design and evaluation of a mobile sensor/actuator network for autonomous animal control," in *Proc. 2007 IEEE IPSN (IPSN'07)*, April 2007.

[2] M. Li and Y. Liu, "Underground structure monitoring with wireless sensor networks," in *Proc. 2007 IEEE IPSN (IPSN'07)*, April 2007.

[3] R. D. Pietro, L. V. Mancini, C. Soriente, A. Spognardi, and G. Tsudik, "Catch me (if you can): Data survival in unattended sensor networks," in *Proc. of 6th Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'08)*, March 2008, pp. 185–194.

[4] D. Zeinalipour-Yazti, V. Kalogeraki, D. Gunopulos, A. Mitra, A. Banerjee, and W. Najjar, "Towards in-situ data storage in sensor databases," in *Proc. of 10th Panhellenic Conference on Informatics (PCI'05), LNCS 3746*, Volos, Greece, 2005, pp. 36–46.

[5] Y. Diao, D. Ganesan, G. Mathur, and P. Shenoy, "Rethinking data management for storage-centric sensor networks," in *Proc. of the Third Biennial Conference on Innovative Data Systems Research (CIDR'07)*, Asilomar, CA, Jan. 2007, pp. 22–31.

[6] J. Girao, D. Westhoff, E. Mykletun, and T. Araki, "Tinypeds: Tiny persistent encrypted data storage in asynchronous wireless sensor networks," *Ad Hoc Networks, Elsevier*, vol. 5, no. 7, pp. 1073–1089, September 2007.

[7] D. Ganesan, B. Greenstein, D. Estrin, J. heidemann, and R. Govindan, "Multiresolution storage and search in sensor networks," *ACM Transactions on Storage*, vol. 1, no. 3, pp. 277–315, August 2005.

[8] F. Armknecht, D. Westhoff, J. Girao, and A. Hessler, "A lifetime-optimized end-to-end encryption scheme for sensor networks allowing in-network processing," *Computer Communications, Elsevier, In Press, Corrected Proof, Available online*, October 2007.

[9] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, "Data-centric storage in sensornets with GHT, a geographic hash table," *Mob. Netw. Appl.*, vol. 8, no. 4, pp. 427–442, 2003.

[10] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin, "Data-centric storage in sensornets," *ACM SIGCOMM, Computer Communications Review*, vol. 33, no. 1, pp. 137–142, January 2003.

[11] S. Nath and A. Kansai, "Flashdb: Dynamic self-tuning database for nand flash," in *Proc. of IPSN07*, April 2007.

[12] A. Banerjee, A. Mitra, W. Najjar, D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos, "Rise-co-s : High performance sensor storage and co-processing architecture," in *Proc. of Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, (SECON'05)*, Santa Clara, California, Sept. 2005, pp. 1 – 12.

[13] A. Mitra, A. Banerjee, W. Najjar, D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos, "High-performance low power sensor platforms featuring gigabyte scale storage," in *Proc. of IEEE/ACM 3rd International Workshop on Measurement, Modeling, and Performance Analysis of Wireless Sensor Networks SenMetrics'2005, (MobiQuitous'05)*, San Diego, CA, 2005.

[14] L. Girod, M. Lukac, V. Trifa, and D. Estrin, "The design and implementation of a self-calibrating distributed acoustic sensing platform," in *Proc. of ACM SenSys (2006)*, 2006.
[15] D. Lymberopoulos and A. Savvides, "Xyz: A motion-enabled, power aware sensor node platform for distributed sensor network applications," in *Proc. of IPSN SPOTS 2005*, April 2005.
[16] R. Pon, M. Batalin, J. Gordon, A. Kansal, D. Liu, L. Shirachi, W. Kaiser, G. Sukhatme, and M. Srivastava, "Networked infomechanical systems: A mobile wireless sensor network platform," in *Proc. of IEEE/ACM IPSN-SPOTS (April 2005)*, 2005.
[17] R.Osrovsky and M.Yung, "How to withsand mobile virus attacks," in *Proc. of PODC*, 1991, pp. 51–59.
[18] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Ultra-low power data storage for sensor networks," in *Proc. of the fifth international conference on Information processing in sensor networks (IPSN'06)*, 2006, pp. 374–381.
[19] A. Perrig, J.A.Stankovic, and D. Wagner, "Secrity in wireless sensor networks," *Communicaion of ACM*, vol. 47, no. 6, pp. 53–57, 2004.
[20] Q. Zhang, T. Yu, and P. Ning, "A framework for identifying compromised nodes in wireless sensor networks," *ACM Transactions in Information and Systems Security (TISSEC)*, 2008.
[21] N. Bhatnagar and E. L. Miller, "Designing a secure reliable file system for sensor networks," in *Proc. of the 2007 ACM Workshop on Storage Security and Survivability (Storage'07)*, Alexandria, Virginia, Oct 2007.
[22] "Intel corporation. intel mote 2." http://www.intel.com/research/.
[23] J. Blomer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, "An xor-based erasure-resilient coding scheme," *ICSI Tech. Report No. TR-95-048*, August 1995.
[24] J. S. Plank and L. Xu, "Optimizing cauchy reed-solomon codes for fault-tolerant network storage applications," in *Proc. of The 5th IEEE International Symposium on Network Computing and Applications (IEEE NCA06)*, Cambridge, MA, July 2006.
[25] J. S. Plank, "Fast galois field arithmetic library in c/c++," *Tech. Report UT-CS-07-593*, 2007.
[26] K. M. Greenan, E. L. Miller, and T. Schwarz, "Analysis and construction of galois fields for efficient storage reliability," *Tech. Report Number SSRC-07-09, UCSC, Storage Systems Research Center*, August 2007.
[27] S. Chessa and P. Maestrini, "Dependable and secure data storage and retrieval in mobile, wireless networks," in *Proc. of DSN'03*.
[28] S. Chessa, R. D. Pietro, and P. Maestrini, "Dependable and secure data storage in wireless ad hoc networks: An assessment of ds2," in *Proc. of WONS 2004*, 2004, pp. 184–198.
[29] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *Proc. of Infocom 2007*, May 2007.
[30] D. Ma and G. Tsudik, "Forward-secure sequential aggregate authentication," in *IEEE Symposium on Security and Privacy(IEEE SP'07)*, Oakland, May 2007, pp. 86–91.
[31] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *Proc. of ISIT'04*, 2004.
[32] S. Jaggi, M. Langberg, S.Katti, T. Ho, D. Katabi, and M. Medard, "Resilient network coding in the presence of byzangtine adversaries," in *Proc. of INFOCOM07*, 2007.
[33] J. Tan and M. Medard, "Secure network coding wiht a cost criterion," in *Proc. of WiOpt'06*, 2006.
[34] L. Lima, M. Medard, and J. Barros, "Random linear newtork coding: a free cipher," in *Proc. of ISIT07*, 2007.
[35] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan, "An efficient signature-based scheme for securing network coding against pullution attacks," in *Proc. of INFOCOM08*, 2008.

Wei Ren is currently an associate professor at School of Computer Science, China University of Geoscience, Wuhan, China. He received his PhD in computer science from Huazhong University of Science and Technology, Wuhan, China, and his MS and BS degrees from University of Science and Technology Beijing, China. He is a senior member of China Computer Federation.

He was a research associate at department of computer science, Hong Kong University of Science and Technology in 2004 and 2005, and a postdoctoral research fellow in school of computer science, University of Nevada Las Vegas in 2006 and 2007. He was with department of electrical and computer engineering, Illinois Institute of Technology in 2007 and 2008. His research interests include network security and information security.

Junge Zhao is currently an associate professor of department of information security, Naval University of Engineering, Wuhan, China. His current research interests include network security and information security.

Yi Ren received his MS degree in computer science from Wuhan University of Technology, Wuhan, China. He is currently a PhD candidate at Information and Communication Technology Department, University of Agder (UIA), Norway. His research interest is network security. He is a student member of IEEE.