2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications

A Lightweight Anonymous On-Demand Routing Scheme in Wireless Sensor Networks

Muhammad Bashir Abdullahi and Guojun Wang*
School of Information Science and Engineering, Central South University
Changsha, Hunan Province, P. R. China, 410083
*Correspondence to: csgjwang@mail.csu.edu.cn

***Abstract*—In wireless sensor networks, all communications between any two sensor nodes occur in a shared and open wireless medium, which makes it easy for an adversary to eavesdrop on every message transmitted. While message encryption can protect the content exchanged between the nodes, routing information may reveal the identities of the communicating nodes and their relationships. For this purpose, the existing schemes used an onion packet, random walk, or storage migration to provide anonymity. These schemes incur high computation cost and communications cost. In this paper, we present a lightweight anonymous routing scheme to provide anonymous communications. In contrast to the existing schemes, we first use a Bloom filter to hide the identities of the sensor nodes, and thereafter, generate per-hop pseudo-link identifiers to accomplish routing and data packets forwarding tasks. This scheme not only conceals the node and the link identities in the data packets, but also helps to achieve traffic anonymity due to content analysis. In this paper, we describe not only the mechanisms to efficiently establish source routes and forward data packets anonymously, but also provide its proof of correctness.**

Keywords-*wireless sensor networks, anonymity, Bloom filter*.

I. Introduction

A wireless sensor network (WSN) consists of resource constrained nodes that communicate wirelessly and self-organize into an ad hoc network [1]. These individual sensors have little value; therefore, it takes advantage of the collective efforts of densely deployed nodes to provide a useful interface to the physical world with their data acquisition, and processing capabilities. Some of their application areas often require the deployment of large-scale sensor networks to vast areas, and hence may result in a huge volume of sensor data.

Recently, sensor networks are deployed, and used for mission-critical applications to collect and analyze sensitive data. Most often, these sensor networks operate in hostile and unattended environments. In such sensor networks, sensitive information (e.g., about battlefield) collected and analyzed may be transmitted and/or stored by some designated sensor nodes in the network; where it can be accessed on-demand. Due to lack of tamper resistance, these sensor nodes are vulnerable to security and privacy violations. Therefore, the role played by some sensor nodes in such a scenario, makes them attractive targets of attack and compromise by an adversary. The role such as a storage node, a source of event data, a cluster head, or a sink, could be used as a reference by an adversary to launch its security and anonymity threats. For this purpose, a variety of anonymous routing protocols such as SDAR[2], MASK[3], ARMA[4], ASR[5], ANODR[6], ODAR[10], and TARo [20], were proposed for mobile ad hoc networks that are closely related to WSNs. The objectives of these protocols include amongst others: to prevent an adversary from learning the identities of the communicating nodes, to thwart the ability of an adversary to trace a data flow either back to the sender, or forward to the receiver, and to guarantee the inability of an adversary to discover the real identities of the local transmitters. However, many existing protocols employed some expensive computation techniques that put stringent requirements on the use of memory, energy consumption, and computational power, which may not be appropriate for resource-constrained WSNs.

In this paper, we propose a lightweight anonymous on-demand routing scheme (LANDER, for short), which first uses Bloom filter [7] to hide the identities of the sensor nodes en route, and thereafter generates per-hop pseudo-link identifiers based on the elliptic curve cryptosystem for accomplishing private information exchange in WSNs. The motivation of this scheme is to construct an energy efficient routing scheme that will prevent an adversary from learning the identities of the sender and the receiver of a data packet, guarantee the inability of an adversary to trace a data flow either back to the sender, or forward to the receiver, as well as ensuring the anonymity of the nodes en route. In contrast to the existing schemes, our scheme is not based on the generation of an onion packet, random walk, or storage migration; rather on the use of authenticated per-hop pseudo-link identifiers combined with Bloom filter. This scheme not only conceals the node and the link identities in data packets, but also helps to achieve traffic anonymity due to content analysis. Furthermore, this scheme employs the concept of reputation-based and trust in order to enhance the security of the data against internal attacks and optimize its reliability, thereby allowing only reliable nodes to participate in all the routing, and packet forwarding activities. In addition, this scheme localizes misbehaving nodes.

The rest of this paper is structured as follows: In Section II, we describe the related work. The models and assumptions under which our scheme operates are presented in Section III. Detailed description of our proposed scheme is presented in Section IV. Section V presents the security and anonymity analyzes of our scheme, and in Section VI the performance analysis of our scheme is discussed. Section VII gives the concluding remarks and the future plan.

II. Related Work

Recently, security and privacy issues in sensor networks have become a major research focus. The problem of source location privacy was introduced and first studied by Chaum [8] who proposed a mixnet to conceal information on a data source. Onion routing was proposed by Reed *et al.* [9] to provide source location privacy in public computer networks. Several protocols were designed especially for WSNs, some of which attempt to provide identity anonymity, route anonymity, and location privacy. In this section, we review some of them.

Subramanian *et al.* proposed Elliptic curve based privacy scheme [11] to conceal data types of reported sensor data and user queries by requiring each node to transform its sensed data (i.e. both the data type and the contents) using a unique secret in such a way that will not disrupt normal query processing at the storage nodes. This scheme introduced some special nodes called anonymizers between nodes and storage nodes that transform the reporting node IDs and the data types - as they received data packets for forwarding - in order to break the potential collusion that may likely occur between the compromised nodes and the storage nodes. Obviously, the transformation scheme ensures the privacy of data and the links but requires the pre-knowledge of the whole routing path from the nodes to the storage nodes. If there is a change in the routing path during the forwarding of data due to some reasons, then the normal query processing will be disrupted at the storage node.

Pan and Boppana proposed anonymous communication protocol [12] to protect actual reporting source nodes location against both local and global adversary by requiring that each node to transmit a fixed-size packet exactly once in each active period regardless of whether it has data to send or not. Destination controlled anonymous routing protocol scheme (DCARPS) is proposed by Nezhad *et al.* [13], in which routing anonymity is based on label switching and onion encryption. In that scheme, a sink is responsible for all routing decisions and thus has the global view of the network topology. The sink constructed a set of shortest path trees between each node and itself. The sink assigned two labels: incoming and outgoing to each node of the tree for uplink and downlink paths, and one label to the leaf nodes and itself. Then, each node of the tree only forwards data packets with labels matching either its downlink or uplink label. Onion encryption is used to change the packet appearance hop-by-hop.

Anonymous path routing (APR) protocol for WSNs is proposed by Jiang *et al.* [14]. This scheme conceals node identity in the single-hop communications and ensures anonymous multi-hop routing. In single-hop communications, the pseudonyms are changed on the per-message basis and the change is based on the mutually shared keys, the sequence numbers and the hash function. In addition, shared keys are updated on the per-message basis. In multi-hop communications, the APR is able to create anonymous paths between any two nodes in the network that initially share the secret key. The anonymous paths are created by flooding the routing request with hidden recipient ID, and labeled with random PathID. Messages are then routed based on the PathID without the source or the destination field; just the pseudonyms are used for anonymous single-hop communications. Intermediate nodes thus only know the preceding and the following node on the path.

In contrast, our approach is similar in concept to the ANODR [6], where each forwarding sensor node adds an encrypted layer to the route request message like an onion without source and destination nodes necessarily knowing the identities of the intermediate nodes. However, we employ Bloom filter-based routing as in ODAR [10], instead of onion routing to provide node, link and path anonymities.

III. Models and Assumptions

A. System Model

We consider a large-scale WSN with densely deployed sensor nodes. The sensing area is divided into some 2D logical regions or grids (called *rendezvous regions*), each with a unique identification [15]. The division is done in such a way that nodes in any two adjacent grids can hear and communicate with each other directly. That is, they are within each other's transmission range. The link between any two nodes is bi-directional. Sensor nodes have overlapping sensing ranges, and events are detected by multiple nodes. Events are all within a fixed region or grid of the network, and not a whole network phenomenon. In other words, grid is the minimum unit for detecting events (referred to as *detection grid*) and for storing event-data (referred to as *storage grid*).

A *trusted mobile sink* (MS), who works as a network controller, will be visiting the network periodically to collect accumulated event-data, and to re-initialize the secret keys as well as to reset the round counters. We assume that all actions in each region or grid are coordinated by a *trusted designated sender* (TDS) or a *receiver* (TDR). The TDS performs the following functions in addition to the functions of a grid coordinator: (i) it collects and aggregates all the sensed data from its grid before it forwards it to the storage grid; and (ii) it also broadcasts beacon packets for building routing structures. Similarly, the TDR (i) receives and distributes event-data value among grid members for storage; (ii) it collects and aggregates all the sensed data from its grid to answer queries from *MS*. To ensure that the load of being a TDS or TDR is rotated among all trusted nodes, election is performed every multiple rounds of data collection. Sensor nodes are stationary and aware of their own location, and a corresponding grid in which they are deployed either through their GPS signal, or a secure localization scheme such as triangulation or trilateration [16].

B. Trust Requirements

Since an *MS* serves as an interface between a WSN and the outside world, compromising it can render the entire network useless. For this reason, we assume that the *MS* is sufficiently powerful to defend itself against security threats. Thus, it is trustworthy, in the sense that it can be trusted if necessary, and is assumed to behave correctly.

We assume that any sensor node elected as either a TDS or TDR is an honest node that does not disclose any information in its possession, and follows all protocols. In each logical grid, the TDS or TDR classifies its grid members into three trust level based groups: lowest, medium, and high, similar to what is described in SDAR [2]. The TDS assigned trust levels based on its direct interactions with each member as well as indirect interactions through its one-hop neighbors based on the exchanged trust information [17]. The TDS computes two different special keys for medium (MTK) and high trust levels (HTK), and share them with the grid members appropriately. All members in the high trust level have both the keys (HTK and MTK); whereas, the members in the medium trust level have only the key for their level, MTK. The members in the lowest trust level only share the general grid key. The HTK and MTK are established in order to secure the broadcast messages that will be sent by a sensor node (or TDS) to its neighbors who are members in a certain trust level.

C. Attack Model

We consider the following categories of attacks:

- *Passive attack*: An adversary passively eavesdrops on the packet transmissions in the network and analyzes packet traffic patterns in order to get valuable information about the communicating nodes and their relationships. This kind of valuable information may include packet source identity and/or location, receiver identity and/or location, and timing of events.
- *Replay routing information attack*: In this attack, the goal of an adversary is to identify the repeated pattern of packet forwarding by replaying old routining information packets or messages repeatedly.

IV. The Proposed Scheme

In this section, we first introduce the bootstrapping and data aggregation phase of our proposed scheme. Then we describe our lightweight and secure anonymous on-demand routing scheme in wireless sensor networks.

Initially the *MS* selects an elliptic curve E defined over a finite field $GF(p)$ of size p, where p is a large prime number and a system base point G ($G \neq O$, where O is a point at infinity) of order q, where q is also a large prime number on that curve. The *MS* then chooses a CCA-secure symmetric block cipher encryption scheme, *Encrypt* and *Decrypt*, a key derivation function, H, of length η and Elliptic curve domain parameters $T = (p, a, b, G, q, h)$ [18]. Then, the *MS* generates its public-private key pair by choosing at random an integer $pvK_{MS} \in GF(q)$ and computes $pbK_{MS} = pvK_{MS}G$. The *MS* then makes all the parameters (E, G, q, pbK_{MS}) public and conceals pvK_{MS} as its private key. Each sensor node is preloaded with a static public-private key (pbK_{s_i}, pvK_{s_i}) pair.

A. Quorum Key Splitting Scheme

QKS is a threshold cryptographic technique that splits a key used for encryption into fragments and distributes those fragments amongst different sensor nodes in a grid. Thus, it mitigates the costs of high availability for the fragments as only l-of-n fragments are collectively required to reconstruct the original key. For each *event-type*, $evtK(j)$, the *MS* constructs a key to be used for encryption and data-location mapping in each grid.

Key Setup: The *MS* chooses uniformly at random an integer $w \in GF(q)$ and computes $X = h(evtK(j)\|t)wG$. Then, the *MS* computes $Y = wpbK_{MS}$ as an implied shared secret value and finally computes key $K = H(X\|pbK_{MS}\|Y)$.

Key Share Generation: For a given key K, to be used in the detection grid $srcG(j)$ $(j = 1, \cdots, m)$ for encryption and secure data-location mapping, the *MS* chooses a d-dimensional vector $v = (K, v_2, \cdots, v_d)$ randomly from $\mathbb{K}^d$ such that $v \bullet \psi(D) = K$; where the *dot* operation is the inner product modulo q.

Key Share Distribution: For each sensor node s_i $(i = 1, \cdots, n)$ in the detection grid $srcG(j)$, the *MS* computes $k_i = v \bullet \psi(s_i)$ and chooses two random numbers $\beta_i \neq 0$ and γ_i both in $GF(q)$ and sends the share pair (k_i, γ_i) to the sensor node s_i over a secured channel. The *MS* then computes $k_i + \beta_i\gamma_i = \tau_i$ and sends to the TDS of detection grid $srcG(j)$, the vector set (β_i, τ_i) as the *check vectors* [19].

Key Reconstruction: To reconstruct the consensus key K, every sensor node s_i $(i = 1, \cdots, n)$ in the detection grid $srcG(j)$ forward its key share pair (k_i, γ_i) to the grid *TDS*. Let $Qs = \{s_1, s_2, \cdots, s_l\} \in \Gamma$ be a qualified subset of the sensor nodes and $\psi(D) = \sum_{i=1}^{l} \lambda_i \bullet \psi(s_i)$, where $\lambda_i \in GF(q)$. The *TDS* verifies each key share pair (k_i, γ_i) received for correctness using the check vector as $k_i + \beta_i\gamma_i = \tau_i$ and accepts when appropriate. Note that the *TDS* has no information about k_i from its check vector, and then it computes the QKS (if at least all the shares of the members of the Qs were received and accepted) as follows:

$$\begin{aligned}\sum_{i=1}^{n} \lambda_i k_i &= \sum_{i=1}^{n} \lambda_i (v \bullet \psi(s_i)) \\ &= v \bullet \sum_{i=1}^{n} \lambda_i \psi(s_i) = v \bullet \psi(D) = K \qquad (1)\end{aligned}$$

Data Encryption: To encrypt a message m using the consensus key K or a pairwise shared key, the TDS or sensor node performs $Encrypt(K, M)$. This is achieved by first choosing a non-repeating nonce $Nc \in \{0, 1\}^*$, which can be a monotonically increasing counter and then computes the ciphertext $C = [M, \tau] = ocbEncrypt(K, Nc, A, m)$, where A is an associated data [22].

Data Decryption: The intended recipient performs $Decrypt(K, C)$ to obtain the original message m that was encrypted using a consensus key or a pairwise shared key. This is achieved by using the corresponding decryption key to compute $m = ocbDecrypt(K, Nc, A, C)$, if it returns a different tag τ, intended recipient stops and rejects the ciphertext. Otherwise, it accepts m as the original message.

B. Designated Coordinators Key System

For each session t, the *MS* also generates a key pair to be shared by all trusted designated grid coordinators. This key will be used during initial anonymous route establishment phase between the two grids (i.e. detection grid $srcG(i)$ and storage grid $dstG(i)$). For this purpose, the *MS* generates n random integers $x_1, \cdots, x_n \in GF(q)$, where n is the number of the grids. For all $srcG(i), dstG(i) \in sdG$, the *MS* computes the secret key $x_t = \sum_{i=1}^{n} h_i(sdG(i))x_i$ and the public key $W_t = x_tG$. The *MS* sends the key pair (W_t, x_t) to the individual trusted designated grid coordinators through a secure channel.

C. Data Aggregation and Mapping

Intuitively, the intra-grid neighbor discovery process, and data forwarding process between the sensor nodes and the grid coordinator are performed based on the LANDER scheme described in Section IV-E. When a grid $srcG(i)$ detects an event $evtD$, all the nodes forward their sensed data piggy-backed with the partial key share to the *TDS*. This is to reduce the communication overhead. The *TDS* aggregates all the data collected based on the RDAT protocol [17], and verifies each key share received from each sensor node s_i for correctness using the *check vector*. Then, it reconstructs the consensus key in order to map the $evtK_i = (srcG_i, t, (srcL_r, srcL_c))$, to the storage grid $dstGi$ location $(dstL_r, dstL_c)$, as described in [15]. Let N denotes the number of grids in the sensor field, N_r and N_c denote the number of rows and the number of columns, respectively. Every grid in sdG is uniquely identified with (x, y), where $0 \leq x \leq N_r - 1$ and $0 \leq y \leq N_c - 1$.

The TDS computes data-location mapping using the consensus key K and the encrypted *event-type* as follows:

- The encrypted *event-type* $\alpha_{e,i} = Encrypt(K, evtK_i)$
- Compute $(x, y) = h(\alpha_{e,i})$

$$\begin{aligned} dstL_r &= h(0\|K\|\alpha_{e,i})\text{mod}(N_r) \\ dstL_c &= h(1\|K\|\alpha_{e,i})\text{mod}(N_c) \end{aligned} \quad (2)$$

where $dstL_{r,i}$ and $dstL_{c,i}$ is the x-coordinate and y-coordinate of the storage grid $dstG_i$, respectively. Note that data-location mapping can be computed based on the temporal attribute of the queries, which may result in dynamic hashed locations over time. Using the QKS scheme the set of hashed locations changes over time due to changes in the consensus key used per session.

- Compute $encD_i = Encrypt(K, evtD_i)$

The Encryption is based on the authenticated-encryption mode known as offset codebook encryption (OCB) mode [22]. This encryption mode provides both secrecy and authentication of the data in only one pass of the block cipher.

- Delete the reconstructed consensus key K from its memory.

D. Bloom Filter Scheme

This section describes the construction of the Bloom filter [7] used in establishing a secure anonymous routing scheme in the next section. The TDS initializes the Bloom filter during route establishment by inserting its information and the information of the destination grid. When a TDS (resp. a sensor node s_i), has data to send (resp. forward) for storage, it first chooses uniformly at random an integer $eSk_s \in GF(2^q)$ to compute $ePk_s = eSk_sG$ as its ephemeral public-private key pair. Then, it generates k hash functions to insert its information into the Bloom filter as follows:

$$hashBF_i(s) = h(eSk_s, (ID_s\|Nc_s\|msgID\|i)), \quad (3)$$

where $1 \leq i \leq k$ is the hash value index, eSk_s is the ephemeral private key of the node s, ID_s is the identifier of the node s, Nc_s is the nonce generated by node s, and $msgID$ is the message identifier. The information of the destination grid inserted by the TDS is:

$$\begin{aligned} &hashBF_i(dstG_i) = \\ &\qquad h(W_t, (dstID\|(dstL_{r,i}, dstL_{c,i})\|msgID\|i)), \end{aligned} \quad (4)$$

where $(dstL_{r,i}, dstL_{c,i})$ is the computed storage grid location coordinates. We note that, here, a false positive may happen due to the probabilistic nature of the Bloom filter [7], but only with a very small (negligible) probability when appropriate parameters are chosen, as we will explain later.

E. Lightweight Anonymous On-Demand Routing Scheme

This section describes the LANDER scheme, which consists of three phases namely: the anonymous route discovery phase, the anonymous route reply phase and the anonymous data forwarding phase.

The anonymous route discovery phase: The TDS broadcasts a route request message locally (i.e. intra-grid broadcast) to its neighbors within wireless transmission range to establish anonymous route when there is need to store event data generated from its grid. The TDS computes an anonymous route request ($LRReq$) packet of the format:

$$\begin{aligned} &LRReq, msgID, dstBF, Treq_s, \\ &\qquad\qquad dTrap, Psm_s, mRn_s \end{aligned} \quad (5)$$

The unique identifier of the packet is $msgID = h(evtK_i, srcG_i, t)$, which is computed using the encrypted *event-type*, the source grid identifier, and the current session. The $dstBF$ is a Bloom filter that contains the information of the initiator and the final destination grid of the request, the sensor nodes en route from the source of the packet to the destination. The trust requirement set by the source node denoted by $Treq_s$ could be high, medium, or low. The purpose of the trust requirement is to ensure that only the trustworthy intermediate nodes are involved in the route construction and subsequent data forwarding activities between the source and the destination. The modified random nonce $mRn_s = Nc_s \oplus hashTK$ will be used to authenticate the nodes that satisfies $Treq_s$. The hash of the key for the trust level specified in the $Treq_s$ is denoted as $hashTK$.

$$dTrap = Encrypt\,(W_t, (dstID, Psm_s, Nc_s)) \quad (6)$$

The $dTrap$ above contains secret information for the intended TDR, and Psm_s and Nc_s are the TDS's pseudonym and nonce generated, respectively. In this phase, the pseudonym and nonce serve as an authentication request from the sender to the receivers of the message that fulfills the trust requirement in the packet header. To compute a pseudonym, each sensor node takes the hash of its static public key, a generated nonce, and the current session as follows: $Psm_i = H\left(pbK_i, \|Nc_i\|t\right)$.

When an intermediate node s_i receives an $LRReq$ packet within a certain time window, it first checks whether the $msgID$ from a certain downlink node has already been recorded in its route table. The node simply discards the packet if the packet ID with the corresponding downlink ID exists and stops. Otherwise, it checks again whether it satisfies the $Treq_s$ in the packet header of the source node. If it is true, then node s_i encodes its information into the $dstBF$, generates its Nc_i and $hashTK$. Then, it replaces the previous Psm_{i-1} and mRn_{i-1} with its own Psm_i and mRn_i, respectively. The sensor node s_i records the $msgID$ and the downlink node's Psm_{i-1} and Nc_{i-1} into its anonymous route table as depicted in Table I, and then broadcasts the modified packet locally to its neighbors. This is because the node s_i may receive several such $LRReq$ through different links.

When a gateway node (similar to a flooder [15]) in an adjacent grid receives the packet, it performs the same operations as explained earlier, and then broadcasts it locally in its grid.

TABLE I
ANONYMOUS ROUTE TABLE

msgID	DlinkID	DLSkey	UlinkID	ULSkey
msgID-1	(Psm_{i-1}, Nc_{i-1})	$\cdots$	$\cdots$	$\cdots$

When a TDR receives the $LRReq$ packet within a certain time window, it first checks whether the $msgID$ from a certain downlink node has already been recorded in its route table. The TDR simply discards the packet if the packet ID with the corresponding downlink ID exists and stops. Otherwise, it checks whether its grid is the final destination by hashing its grid information into the embedded $dstBF$. If it is true, then it records the information into its route table as depicted in Table I, and rebroadcasts the $LRReq$ once. The destined TDR verifies the $dTrap$ embedded in the $LRReq$ packet and then initiates route reply $LRRep$. The rebroadcast by the TDR is to fool an adversary observing the packet transmission in order to identify the final receiver. Otherwise, the TDR rebroadcasts the packet locally only. These processes continue until the packet finally reaches the destined grid.

The anonymous route reply phase: The destined TDR constructs a route reply $LRRep$ packet with the following format and then forwards it in reverse paths towards the source grid where the $LRReq$ packet was initiated.

$$LRRep, msgID, dstBF, \{Psm_d, Nc_d\}_{Nc_n}, sTrap \quad (7)$$

The TDR lookups the $DlinkID$ field of its anonymous route table entry for the nonce(s) extracted and uses them to encrypt the $sTrap$ for the TDS and its pseudonym and a nonce as $\{Psm_d, Nc_d\}_{Nc_n}$ for the $LRRep$ uplink-node (formerly the $LRReq$ downlink-node) accordingly. This signifies an agreement from the TDR to achieve mutual authentication with its qualified uplink nodes for this task. For example, if node i is on the routing path of node j's message and that it will help to forward the message to the intended destination, then node i is the uplink-node of node j. Conversely, node j is the downlink-node of node i.

$$sTrap = \{dstID, dstL, Psm_d, Nc_d\}_{Nc_s} \quad (8)$$

The $sTrap$ is the proof generated by the TDR that it has successfully opened the $dTrap$ embedded in the $LRReq$ packet. The contents of the proof include: the destination grid identifier, its location coordinates $dstL = (dstL_{r,i}, dstL_{c,i})$, the pseudonym, and the nonce generated by the current grid coordinator, TDR. When an intermediate sensor node s_i on the reverse path receives the $LRRep$ packet within a certain time window, it first checks whether the $msgID$ from a certain uplink node has already been recorded in its route table. The node simply discards the packet if the packet ID with the corresponding uplink ID exists and stops. Otherwise, it checks again the $dstBF$ whether its information is encoded into it. The node simply discards the packet if the test fails and stops. Otherwise, it checks again whether it is the intended receiver using its Nc_{i-1} to decrypt $\{Psm_i, Nc_i\}_{Nc_{i-1}}$. The node simply discards the packet if the decryption fails and stops. Otherwise, the intermediate sensor node s_i lookups the $DlinkID$ field of its anonymous route table entry for the nonce(s) extracted and uses them to encrypt its pseudonym and a nonce as $\{Psm_{i-1}, Nc_{i-1}\}_{Nc_{i-2}}$ for the $LRRep$ uplink-node(s) (formerly the $LRReq$ downlink-node) accordingly. This signifies an agreement from the sensor node s_i to achieve mutual authentication with its qualified uplink nodes for this task. A neighbor node that does not satisfies the trust requirement set in $Treq_s$; cannot be able to extract Nc_i from mRn_i, because it has not access to $hashTK$.

TABLE II
ANONYMOUS FORWARDING ROUTE TABLE

msgID	DlinkID	DLSkey	UlinkID	ULSkey
msgID-2	$L_{t,i\leftrightarrow j}$	$K_{t,i\rightarrow j}$	$L_{t,i\leftrightarrow k}$	$K_{t,i\rightarrow k}$

In addition, the intermediate sensor node s_i will implicitly authenticate its qualified immediate neighbors (i.e., those that successfully extracted the original nonce, Nc_i) anonymously by establishing a pairwise session keys, and link identifiers accordingly. The established key and link identifier exclusively belong to the two end sensor nodes. If a sensor node has more than one authenticated neighbors, then it will have alternative paths, or links to use whenever one neighbor node is unreachable due to some reasons.

Similarly, once the intended TDS receives the $LRRep$ packet within a certain time window, it first forwards it ahead in order to fool an adversary observing the packet transmission, and then checks the $dstBF$ for the encoding of its information. If it is true, then the TDS decrypts the embedded proof $sTrap$, by the destined TDR using its Nc_s. The TDS compares $dstID$ and the grid location coordinates embedded in the $sTrap$ to the initially generated coordinate values $dstL$. If they are accurate, then the TDS uses the TDR's pseudonym, and the nonce included in the $LRRep$ packet to calculate a pairwise session key, and link identifier as explained above. The TDS and TDR will implicitly authenticate each other and their qualified immediate neighbors by establishing a pairwise session keys, and link identifiers for either party accordingly. This method of authentication helps to reduce operating complexity and minimize power consumption. Every sensor node update its anonymous route table accordingly as depicted in Table II.

Every sensor node computes the pairwise session key $LSKey$ and link identifier $linkID$, respectively as follows:

$$s_i\text{: } K_{t,i\to j} = H\left(Psm_i \oplus Psm_j \| Nc_i \times Nc_j \times G\right)$$
$$s_j\text{: } K_{t,j\to i} = H\left(Psm_j \oplus Psm_i \| Nc_j \times Nc_i \times G\right) \quad (9)$$
$$s_i, s_j\text{: } L_{t,i\leftrightarrow j} = H\left(Psm_i \| Nc_j \| Psm_j \| Nc_i \| t\right) \quad (10)$$

where G denotes the elliptic curve system base point and t denotes the current session. Each node en route from the source to the destination, generates two pairs of $\langle LSKey, linkID\rangle$ for its uplink-nodes, and downlink-nodes. Each node then, updates its route mapping table. Due to the difference in the pseudonyms and nonces for each pair of sensor nodes, and the use of a collision-resistance hash function, the set of pairwise shared secrets $\langle LSKey, linkID\rangle$ will be unique for each link (incoming and outgoing). The $linkID$ will be used in the session t to identify the packets transmitted between the node s_i and the node s_j, and the $LSKey$ can be used to encrypt and protect the integrity or authenticate the contents of packets, if need arises [3]. The unique feature of this individual generated $LSKey$ is that, one pair of the key controls the encryption procedure, while the other pair controls the decryption procedure [18]. Thus, it is hard for an adversary who does not possess the other party's secret key to recover the plaintext from its ciphertext. Now, the anonymous route discovery is completed.

The anonymous data forwarding phase: After the anonymous route discovery phase, the TDS performs the following:

- Execute **put**$(\alpha_{e,i}, encD_i, Q_i)$

Where $encD_i$ and Q_i is the encrypted data and the QoS in the selection of the sensor nodes required to store the data $encD_i$, respectively [21].

- Generate a packet
  $P_{store} = \langle(dstL_{r,i}, dstL_{c,i}), \langle\alpha_{e,i}, encD_i, Q_i\rangle\rangle$

The TDS lookups its forwarding route table entry and picks the next hop $linkID$ towards the intended destination and that of the destination. The TDS forwards the packet to the next hop via the route to the destination grid with the following format:

$$L_{t,s\leftrightarrow 1}, \left\{DATA, dstBF, \{P_{store}\}_{K_{t,s\to d}}\right\}_{K_{t,s\to 1}} \quad (11)$$

Where $\langle K_{t,s\to 1}, L_{t,s\leftrightarrow 1}\rangle$ is the pairwise shared secret between the source node TDS and the next uplink-node along the path toward the intended destination grid. $DATA$ is the message type. The generated shared link between the source and the destination concatenated with the destination grid location coordinates are encoded into the Bloom filter as explained earlier: $dstBF \leftarrow H\left(K_{t,s\to d}, L_{t,s\leftrightarrow d} \| (dstL_{r,i}, dstL_{c,i})\right)$.

The destination bound data is encrypted using the generated $LSKey$, $K_{t,s\to d}$ implicitly shared with the destination grid coordinator, TDR. The other contents of the packet are the same as explained earlier.

Upon receiving the data packet, the uplink intermediate node sharing the $linkID$ will decrypt the packet using the corresponding $LSKey$ and then lookup its forwarding route table to pick the $linkID$ for the next uplink-node matching the downlink $linkID$, encrypt the packet using the corresponding $LSKey$, change the $linkID$ and then forward it to the next node as follows:

$$L_{t,i\leftrightarrow j}, \left\{DATA, dstBF, \{P_{store}\}_{K_{t,s\to d}}\right\}_{K_{t,i\to j}} \quad (12)$$

This process continues until the packet reaches a grid coordinator, TDR, who will now verify the $dstBF$ whether the packet is bound for its grid. If it is true, the TDR will decrypt the packet using its corresponding $LSKey$, $K_{t,d\to s}$ and then store the encrypted event-type and event data $(\alpha_{e,i}, encD_i)$ pair according to the requirements prescribed in Q_i depending on the application. Otherwise, the TDR replaces the downlink $linkID$ with the matching uplink $linkID$ in its forwarding route table, encrypt the packet using the corresponding $LSKey$ and then forward it to the next node.

F. Determining Optimal Bloom Filter Size

Since the hash functions used to insert an element into a Bloom filter are independent, there is possibilty of collisions in their outputs, resulting to false positives for some elements. Therefore, it is important to make sure that the false positive rate is as small as possible. Assume that m is the number of bits in the array and that a hash function selects each array position with equal probability. If n elements have been inserted using the k hash functions, the probability that a certain bit is still 0 is $(1 - 1/m)^{kn}$. Thus, the probability that a certain bit is set to 1, which is also termed as expected fill ratio EFR is $1 - (1 - 1/m)^{kn}$ [7]. Therefore, the false positive rate FPR is often given as $FPR = (1-(1-\frac{1}{m})^{kn})^k$, from Taylor series expansion, we have $FPR \approx (1 - e^{\frac{kn}{m}})^k$.

From the above analysis, it shows that the false positive rate FPR decreases as m and k increases, and increases as n increases. For any given FPR and m, n is maximized by making EFR = 1/2, regardless of FPR or m [7]. Thus, we have $k = (m \ln 2)/n$, which gives the $FPR = 2^{-k} \approx (0.6185)^k$. Substituting for the value of k, we can obtain the probability FPR as a function of only two parameters: m and n. This in turn allows us to easily calculate the desired size of the Bloom filter as a function of the false positive rate FPR and the expected number of network sensor nodes n.

V. Security and Anonymity Analysis

In this section we analyze the security and anonymity of our proposed LANDER scheme.

***Theorem** 1: The LANDER scheme is secure against the passive and replay attacks.*

Proof: The LANDER is resilient to Passive and replay attacks. An adversary that passively eavesdrops on the packet transmissions and analyzes packet traffic patterns based on its length, type, counting or size will find it difficult to identify and correlate packet as it flows across different links in the network. This is because the packet size is randomized and thus appears different at each hop due to changes in the link identifier and per-hop link encryption using the established pairwise shared secret $\langle LSKey, linkID\rangle$ pair. Similarly, an adversary that engages in replay attack is frustrated by periodically changing the link identifiers for the same flow at each hop. Because the replay attack involves resending of old packets that use correct link identifiers. ■

***Theorem** 2: The LANDER scheme ensures that: (a) no one knows the real identities of the source and the destination, except themselves; (b) the source and the destination have no information about the real identities of intermediate nodes en route.*

Proof: The LANDER scheme ensures the identity privacy of the sender, receiver and the intermediate nodes. In LANDER, during the route discovery processes, the true node identities are hidden by using pseudonyms. Secondly, forwarding node identities are hidden by securely hashing the node identity and the related information into Bloom filters of the source routes. No single node or any adversary can verify the membership of a particular node, except itself. This provides local source and receiver anonymity, and the relationship between them. During the route reply phase, two neighbor nodes authenticated each other and established a pairwise shared secret $\langle LSKey, linkID\rangle$ without revealing their true identifiers. During data forwarding phase, true identities were not used rather the shared link identifiers. For an adversary (local or global) to identify a link identifier during transmission process, it is very difficult to associate it to any node. Thus, LANDER achieves unlinkability. ■

***Theorem** 3: The LANDER scheme ensures that: (a) no one knows the exact location of the source or the destination, except themselves; (b) other nodes, typically intermediate nodes en route, have no information about their distance, i.e. the number of hops, from either the source or the destination.*

Proof: The LANDER scheme ensures location privacy of the source, the destination and the intermediate nodes. An adversary violates location privacy using either packet header information or tracing packet flows to the origin or destination. In LANDER the source, the intermediate forwarding nodes and the destination information were concealed in a secure source Bloom filter. It is only the nodes themselves can verify their membership in the Bloom filter of the source route. Therefore, using packet header information to identify the location of a packet source or destination will not reveal any important information to the adversary. ■

***Theorem** 4: The LANDER scheme guarantees that: (a) adversaries, either en route or out of the route, cannot trace a packet flow back to its source or destination; (b) for adversaries not in the route, they have no information on any part of the route.*

Proof: The LANDER scheme guaranteed the path anonymity between the source of a packet and the destination of that packet. An adversary violates path anonymity through packet traffic analysis, which can be either content analysis attack or timing analysis attack. In the content analysis attack, an adversary correlates common information; such as sequence number, length, size of the packet, etc., among the eavesdropped packets on successive links and therefore, identify the path of traffic flows between the source and the destination. In onion routing schemes, an adversary can easily estimate the hop distance to the source by examine the length of the onion as it grows in size with the hop distance from the source. The LANDER randomizes the packet size and thus appears different at each hop due to changes in the link identifiers and per-hop link encryption using the established pairwise shared secret $\langle LSKey, linkID\rangle$ pair. Therefore, matching packets for common information will not avail to any adversary; the end-to-end path of transmission due to dynamically changing of the per hop packet information. In timing analysis attack, an adversary correlates the time of packet transmission on successive links; by observing the transmission in terms of frequency and locations. It can easily guess that two successive transmissions belong to the same communication flows. Therefore, the adversary can use it to identify the path of traffic flows between the source and the destination. The LANDER employs trustworthy nodes in all routing processes, so packets can be forwarded at a random interval of time in each successive link. This temporal randomization can increase the effort of an adversary in determining the path of traffic flows between the source and the destination. ■

VI. Performance Analysis

We consider the storage cost, communication, and computation overhead of our LANDER scheme. In our approach, both the broadcast $LRReq$ message, and $LRRep$ message remains a constant size, and the key exchange is placed in the unicast reverse path (route reply). For the same reason, computational overhead is also significantly reduced, as cryptographic operations are only performed in the limited number of intermediate nodes. Assume that all symbols used are elements of a Galois Field $GF(2^q)$, where q = 8 or 16. Let S be the number of nodes en route from source to destination inclusive.

Storage Cost: Each node keeps a neighbor trust table (NTT), and anonymous forwarding route table (AFRT). Therefore, the average storage cost (ASC) is given as:

$$avgSC = \frac{1}{S}\sum_{i=1}^{S}(|NTT|_i + |AFRT|_i)q. \quad (13)$$

Communication Overhead: During the route request phase, each node broadcast an $LRReq$ message, and during the route reply phase each node en route unicast at least n $LRRep$ messages; where n is the average number of qualified uplink nodes. So, the average communication overhead (ACO) is as:

$$avgCO = \frac{1}{S}\sum_{i=1}^{S}(|LRReq|_i + n\,|LRRep|_i)q. \quad (14)$$

Computation Overhead: Every node en route from source to destination during route discovery phase (i) Performs $2{*}k$ hash operations to insert into and verify its information from the Bloom filter, 1 hash operation to generate its pseudonym, and $4{*}n$ hash operations to generate a pairwise key and link identifier for the uplink and the downlink nodes, respectively. (ii) Performs 2 point multiplications to generate a key for uplink and downlink nodes. (iii) Performs n encryption and decryption operations during route reply phase.

In addition to the above overhead, the TDS and the TDR performs k hash operations to insert into and verify the destination's information from the Bloom filter, respectively. The TDS and the TDR performs 1 encryption and decryption operation each. Therefore, the total computation (TC) is:

$$\begin{aligned}TC = S(2H_\alpha^k + H_\beta^1 + 2H_\mu^n + H_\gamma^n + PtM^2)\\ + (S-2)(Enc^n + Dec^n) + Enc^1 + Dec^1. \quad (15)\end{aligned}$$

VII. Conclusion

In this paper, we presented a lightweight anonymous on-demand routing scheme in WSNs (LANDER for short). In the LANDER scheme, after path establishment, each pair of neighbor nodes authenticate each other anonymously, and establish a pseudo-link identifier with an associated pairwise key. This pairwise key and pseudo-link identifier are used for anonymous data forwarding. Unlike the onion routing scheme that placed high cryptographic burden on the two end nodes (i.e., the source node and the destination node), the LANDER scheme is flexible, and ensures load balancing as it shares the burden of the cryptographic operations among the sensor nodes along the path including the two end nodes. The use of a lightweight cryptographic scheme and the use of memory and bandwidth efficient Bloom filter, make the LANDER scheme appropriate for resource constrained WSNs.

Our future work is to implement the LANDER scheme in a grid based data-centric storage architecture and analyze its performance using simulations.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under grant numbers 61073037 and 61103035, and the Ministry of Education Fund for Doctoral Disciplines in Higher Education under grant number 20110162110043.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey," *Computer Networks*: (Elsevier Science), 38(4): 393-422, 2002.

[2] A. Boukerche, K. El-Khatib, L. Xu, and L. Korba, "An Efficient Secure Distributed Anonymous Routing Protocol for Mobile and Wireless Ad Hoc Networks," *Computer Communications*, 28: 1193-1203, 2005.

[3] Y. Zhang, W. Liu, and W. Lou, "Anonymous Communications in Mobile Ad Hoc Networks," *Proc. of Annual IEEE Conference on Computer Communications* (INFOCOM'05), 1940-1951, 2005.

[4] A. Boukerche and Y. Ren, "ARMA: An Efficient Secure Ad Hoc Routing Protocol," *Proc. of the IEEE Global Communications Conference* (GLOBECOM'07), 1268-1272, 2007.

[5] B. Zhu, Z. Wan, M. S. Kankanhalli, F. Bao, and R. H. Deng, "Anonymous Secure Routing in Mobile Ad Hoc Networks," *Proc. of the 29th IEEE International Conference on Local Computer Networks* (LCN'04), 102-108, 2004.

[6] J. Kong, X. Hong and M. Gerla, "An Identity-Free and On-Demand Routing Scheme Against Anonymity Threats in Mobile Ad Hoc Networks," *IEEE Transaction on Mobile Computing*, 6(8): 888-902, 2007.

[7] B. H. Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors," *Communications of the ACM*, 13(7): 422-426, 1970.

[8] D. L. Chaum, "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," *Communications of the ACM*, 24(2): 84-88, 1981.

[9] M. G. Reed, P. F. Syverson, and D. M. Goldschlag, "Anonymous Connections and Onion Routing," *IEEE Journal on Selected Areas in Communications*, 16(4): 482-494, 1998.

[10] L. Bao, R. Chen and D. Sy, "On-Demand Anonymous Routing in Ad Hoc Networks (ODAR)," *Security in Ad Hoc and Sensor Networks* (Computer and Network Security), 3: 137-157, 2008.

[11] N. Subramanian, K. Yang, W. Zhang, and D. Qiao, "ElliPS: A Privacy Preserving Scheme for Sensor Data Storage and Query,"*Proc. of Annual IEEE Conference on Computer Communications* (INFOCOM'09), 936-944, 2009.

[12] P. Pan and R. V. Boppana, "ACP: Anonymous Communication Protocol for Wireless Sensor Networks," *Proc. of the 8th IEEE Consumer Communications and Networking Conference* (CCNC'11), 751-755, 2011.

[13] A. Nezhad, A. Miri, and D. Makrakis, "Location Privacy and Anonymity Preserving Routing for Wireless Sensor Networks,"*Computer Networks*, 52(18): 3433-3452, 2008.

[14] J.-R. Jiang, J.-P. Sheu, C. Tu, and J.-W. Wu, "An Anonymous Path Routing (ARP) Protocol for Wireless Sensor Networks," *Journal of Information Science and Engineering*, 27: 657-680, 2011.

[15] K. Seada and A. Helmy, "Rendezvous Regions: A Scalable Architecture for Service Location and Data-Centric Storage in Large-Scale Wireless Networks," *Proc. of the 18th International Parallel and Distributed Processing Symposium* (IPDPS'04), 218-225, 2004.

[16] S. Capkun and J.-P. Hubaux, "Secure Positioning of Wireless Devices with Application to Sensor Networks,"*Proc. of Annual IEEE Conference on Computer Communications* (INFOCOM'05), 1917-1928, 2005.

[17] S. Ozdemir, "Functional Reputation Based Data Aggregation for Wireless Sensor Networks," *Proc. of the fourth IEEE International Conference on Wireless and Mobile Computing, Networking and Communications* (WiMob'08), 592-597, 2008.

[18] D. R. L. Brown, *Standards for Efficient Cryptography 1(SEC-1)*, Certicom Research. http://www.secg.org/secg_docs.htm.

[19] T. Rabin and M. Ben-Or, "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority,"*Proc. of the 21st Annual ACM Symposium on Theory of Computing* (STOC'89), 73-85, 1989.

[20] J. T. Chen, R. Boreli, V. Sivaraman, "TARo: Trusted Anonymous Routing for MANETs," *Proc. of the Sixth IEEE/IFIP International Symposium on Trusted Computing and Communications (TrustCom-10)*, IEEE Computer Society, 756-762, 2010.

[21] M. Albano, S. Chessa, F. Nidito, and S. Pelagatti, "Dealing with Nonuniformity in Data Centric Storage for Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, 22(8): 1398-1406, 2011.

[22] T. Krovetz and P. Rogaway, "The Software Performance of Authenticated-Encryption Mode,"*Proc. of the 18th International Workshop on Fast Software Encryption*, (FSE 2011), 6733: 306-327, 2011.