An Enhanced Communication Protocol for Anonymity and Location Privacy in WSN

Abdel-shakour Abuzneid, Tarek Sobh, and Miad Faezipour
University of Bridgeport, Bridgeport, Connecticut
Email: {abuzneid, sobh, mfaezipo}@bridgeport.edu

Abstract—**Wireless sensor networks (WSNs) consist of many sensors working as hosts. These sensors can sense a phenomenon and represent it in a form of data. There are many applications for WSNs such as object tracking and monitoring where the objects need protection. Providing an efficient location privacy solution would be challenging to achieve due to the exposed nature of the WSN. The communication protocol needs to provide location privacy measured by anonymity, observability, capture- likelihood and safety period. We extend this work to allow for countermeasures against semi-global and global adversaries. We present a network model that is protected against a sophisticated passive and active attacks using local, semi-global, and global adversaries.**

Keywords—WSN; anonymity; privacy; source location privacy; sink privacy; contextual privacy

I. INTRODUCTION

A wireless sensor is a simple autonomous host device. It can sense a phenomenon, convert the sensed information into data, process the data and then transmit the data to a sink for further analysis. The sensor host is very limited in terms of storage space, cache memory, computing power, communication bandwidth and battery energy [1]. This work focus on monitoring and tracking applications such as tracking animal in the wildlife or a fellow soldier in the field. When the sensor node senses an object, it reports data to the sink through other neighboring sensors. One of the most common applications discussed in literature is the *panda monitoring game* [1]. When a sensor node detects a Panda in a certain area, it should report it via a message to the sink. In order to protect the Panda from adversaries (ADVs), we need to implement in place an efficient source location privacy scheme (SLP). SLP is even more important in military, homeland security, and law enforcement, in addition to many civilian applications [2-4]. In addition, the network need to provide sink location privacy (SinkLP).

II. PROBLEM STATEMENT

Privacy in WSN is typically categorized into two categories: *data privacy* and *contextual privacy* [5]. In this work, we shall focus on using anonymity to provide location privacy which includes SLP and SinkLP. There are many solutions which have been presented to solve the problems of SLP and SinkLP. Recently, Anonymity has become a major concern for some WSN applications. We have identified important literature discussing solutions for anonymity in WSN such as, SAS, Simple Anonymity Scheme [6]; CAS, Cryptographic Anonymity Scheme [6]; HIR, Hashing-Based ID Randomization [7]; APR, Anonymous Path Routing [8]; DCARPS, Destination Controlled Anonymous Routing Protocol for Sensor Nets [2]. None of these solutions provide location privacy against active attacks and multiple colluding adversaries. An important solution against global adversary introduced by Chen et al. [9] called Efficient Anonymous Communication (EAC) which claims providing sender, link and sink anonymity. The solution is light, however, it can easily lose pseudonym synchronization. In addition, it fails to provide SinkLP and it is not secure against traffic rate analysis attack. We know that anonymity is not sufficient to provide end-to-end privacy. There are some solutions based on dummy (fake) data sources where sensors send out dummy packets to other nodes within the network. The rest of this paper is organized as follows: In section 3, we give some preliminaries for the system model, network model, threat model and the traffic model. In section 4, we will introduce our anonymity protocol. In section 5, we will provide security analysis and simulation. In section 6, we will summarize our work and suggest future work.

III. PRELIMINARIES

We assume bi-directional links where two nodes are considered neighbors if they can hear each other [2].The network considers one sink which aggregates sensed data from all sensors. The sink works as an interface for WSN to the wired network [6]. Data packets generated by the sensors are ultimately transferred uplink to the sink, however, it could go through a multi-hop route. Control packets can be sent from the sink, downlink, to the sensors by unicast or broadcast messages. To enhance SinkLP, the sink behaves like any other sensor in the network when it communicates with other sensors to make it absolutely indistinguishable. The WSN runs in three phases: deployment phase, icebreaker phase, and communication phase. We assume that sensors have the ability to obfuscate the addresses at the MAC level header [6]. The network adopts a time synchronization protocol [6]. The WSN will adopt a protocol for *network topology discovery* that allows the sink to view the global topology of the network without revealing its location [2]. The adversary nodes have much stronger power compared to the sensors. An adversary could run both *passive* and *active* attacks. We presume that only few compromised nodes can exist at one time due to the implementation of intrusion detection system (IDS) [10]. We assume a global adversary which can monitor the traffic of the entire network and can determine the node responsible for the initial transmission. We also assume that the adversary is capable of observing sensors' transmissions over extended periods of time. It is, however, not able to break the encryption algorithm or the hash functions adopted in the protocol.

IV. Proposed anonymous model

The communication process is divided into three phases, namely: deployment, icebreaker, and communication phases.

A. Deployment phase:

Prior to actual distribution of the sensors in the field of application, the sensors need to be tested, fully charged, and loaded with the parameters which are needed during the setup phase and then during the communication phase.

B. Icebreaker phase

In this phase, sensors get to know about their location and surroundings. Typically, WSN is considered secure for some short period of time after the deployment of sensors and before the steady communication phase. Zhu et al. [11] presented that WSN has Tmin, which is the shortest time that the adversary needs to be able to compromise a sensor. During this time, the sensors can communicate and exchange all needed information, such as preloaded parameters as well as some calculated parameters. The sink needs to know the location of all the sensors participating in the WSN. Likewise, the sensors need to know their relative locations to the sink and to the neighboring sensors. There are many localization schemes which are proposed in the literature [6]. After the localization process is completed, each sensor Si will know its $hc_{i\leftrightarrow sk}$, the smallest hop-count to the sink. All parameters and terms are listed in TABLE I.

1) Creating Pseudonyms

A sensor needs to use one disposable ID per one transmission which means, the sensor uses any issued pseudonym only once. There are five kinds of transmissions that could happen during the communication phase: (i) multi- hop transmission to the sink, (ii) transmission between two neighbors, (iii) broadcast sent by S_i or the sink, (iv) acknowledgement, and (v) dummy packet broadcast. The process starts by creating a pseudonym for each sensor S_i, we call it for short (SID_i) which is computed using expression (1):

$$SID_i = H(ID_i \oplus \alpha_i) \qquad (1)$$

TABLE I. Reference of all parameters and terms used for the proposed framework.

Notation	Definition
α_i	Random number shared between S_i & SINK
β_i	Random number shared between S_i & neighbors
γ_i	Random number shared between S_i & neighbors
H	Hash function to create pseudonyms and the keys.
$k_{i\leftrightarrow sk}$	Pair-wise key shared between S_i & SINK.
kb_i	Broadcast key for S_i.
kdb_i	Dummy broadcast key for S_i.
N_i	Number of neighboring for S_i.
$hc_{i\leftrightarrow sk}$	Hop-count between S_i & SINK.
SID_i	Pseudonym ID shared between S_i & SINK.
$BSID_i$	Broadcast pseudonym ID.
$\alpha_{i\leftrightarrow j}$	Random value shared between S_i & S_j.
$k_{i\leftrightarrow j}$	Pair-wise key shared between S_i & S_j.
$HSID_{i\leftrightarrow j}$	Pseudonym ID shared between S_i & S_j.
$ASID_i$	ACK pseudonym ID for Si.
$DSID_i$	Dummy broadcast pseudonym ID.
T_i	Table in Si for shared parameters.

S_i calculates the broadcast pseudonym ID ($BSID_i$) according to the following expression:

$$BSID_i = H(ID_i \oplus \beta_i) \qquad (2)$$

Then, S_i calculates the dummy broadcast pseudonym ID ($DSID_i$) according the following expression:

$$DSID_i = H(ID_i \oplus \gamma_i) \qquad (3)$$

S_i should, by now, know its entire neighbors set N_i. A neighbor S_j is a sensor that could receive signal from the S_i and vice versa through one-hop transmission. S_i will send a broadcast discovery message *(DISC)*, to exchange parameters with all one-hop neighbors as below:

$$DISC = TTL \parallel ID_i \parallel k_{i\leftrightarrow sk} \parallel kb_i \parallel kdb_i \parallel \alpha_i \parallel \beta_i \parallel \gamma_i \parallel hc_{i\leftrightarrow sk} \qquad (4)$$

Where TTL=1 for this transmission. S_i will receive also a similar broadcast message from S_j and from all other neighbors. Both S_i and S_j will calculate a new random value ($\alpha_{i\leftrightarrow j}$) according to expressions below:

$$\alpha_{i\leftrightarrow j} = H(ID_i \oplus ID_j) \qquad (5)$$

Both S_i and S_j will calculate also a new pair-wise key $k_{i\leftrightarrow j}$ according to expression below:

$$k_{i\leftrightarrow j} = H(k_{i\leftrightarrow sk} \oplus k_{j\leftrightarrow sk}) \qquad (6)$$

S_i also calculates broadcast pseudonym ID for S_j ($BSID_j$) according to expression (2) since S_i has already received the values of ID_j and β_j through DISC. It also calculates the one-hop pseudonym ID ($HSID_{i\leftrightarrow j}$) shared between S_i and S_j as in the expression below:

$$HSID_{i\leftrightarrow j} = H(\alpha_i \oplus \alpha_j) \qquad (7)$$

Finally, acknowledgement pseudonym ID for S_i ($ASID_i$) will be calculated according to the expression below:

$$ASID_i = H(ID_i) \qquad (8)$$

S_i will have a table T_i which contains the shared values with the neighbors as listed in TABLE II. Thus, we have replaced the ID with *quintuple pseudonyms* to reference the sensor during communication.

2) Deleting security information

After storing all required pseudonyms, parameters and keys in T_i for S_i, it would be the time to delete all unnecessary information for the purpose of security. In addition, it will release some memory storage space [9, 12]. Most importantly, S_i will delete ID_i and $hc_{i\leftrightarrow sk}$, which could be critical information for the adversary. In addition, S_i shall delete discovery messages received earlier from the neighbors.

C. Communication phase

During the communication phase, when sending data to the sink takes place, there are seven operations that continue until the network dies. These operations are: sense and send a packet to a neighbor, forward a packet hop-by-hop, broadcast a real packet, acknowledgement, broadcast a dummy packet, a sensor cancelation, and a new sensor deployment. A sensor can have three roles, in terms of data transmission, during the communication phase:

TABLE II. SHARED VALUES AMONG SENSOR NEIGHBORS. IF S_i HAS N_i NEIGHBORS, THEN T_i WILL HAVE N_i ROWS.

Information in Ti per each neighbor	Tuple for Sj
Shared random number	$\alpha_{i\leftrightarrow j}$
Shared broadcast random number	β_j
Shared dummy random number	γ_j
Shared broadcast key	$BSID_j$
Shared dummy broadcast key	$DSID_j$
Shared one-hop key	$k_{i\leftrightarrow j}$
Current one-hop pseudonym ID	$HSID_{i\leftrightarrow j}$
Link direction	$link_{i\rightarrow j}$
Residual energy level	Δ_j

1) Sensing data

S_i only recognizes itself by its newly calculated pseudonym (SID_i), and the sink can recognize source of the packet through SID_i identity as well. Thus, the SID_i of the source needs to be included in the packet until it reaches to the sink. Consequently, the SID of S_i will be updated after every transmission. When data is sensed and ready to be sent along, S_i needs to select one neighbor to forward the message to. The selection process goes through a probabilistic protocol which guarantees that S_i does not depend all the time on one neighbor for routing privacy and for increasing the lifetime of the WSN. S_i will form the message in the following format:

$$M_{i\rightarrow j} = HSID_{i\leftrightarrow j} \,\|\, E_{ki\rightarrow j}\,(SID_i \,\|\, E_{ki\leftrightarrow sk}\,(D_i)) \,\|\, HMAC_{ki\leftrightarrow sk}\,(SID_i \,\|\, D_i) \qquad (9)$$

Once S_i knows that the packet ($M_{i\rightarrow j}$) is delivered to the sink, it needs to dispose the current pseudonym SID_i and issue a new one for the next transmission as below:

$$SID_i = H(SID_i \oplus \alpha_i) \qquad (10)$$

In addition, both S_i and S_j will dispose the current $HSID_{i\rightarrow j}$ and issue a new one for the next communication between the two sensors as below:

$$HSID_{i\rightarrow j} = H(HSID_{i\rightarrow j} \oplus \alpha_{i\rightarrow j}) \qquad (11)$$

The packet M will then be reformatted by the recipient S_j and again forwarded to the next node, in the path, S_r and so on, until it gets to the sink. If S_j was the sink, then it uses the shared one-hop key between itself and the neighboring sensor, to decrypt the data and gets the SID_i which the it can use to recognize S_i. Thus, the sink will be able to recognize the originator of this message. Only at this point of time, it can update the value of SID_i of S_i. It also gets the data (D_i) which it can read after decrypting it using $k_{i\leftrightarrow sk}$.

2) Forwarding data

When S_i sends the message one-hop uplink to the neighbor S_j, S_j needs to forward the message to another intermediary node. Upon receiving $M_{i\rightarrow j}$, S_j will match $HSID_{i\rightarrow j}$ in its table, T_j. If there is no match, then the message will be dropped immediately. If it matches, then the message will be decrypt using $k_{i\rightarrow j}$. The message will be forwarded to S_r after M is reformatted as below:

$$M_{j\rightarrow r} = HSID_{j\leftrightarrow r} \,\|\, E_{kj\rightarrow r}\,(SID_i \,\|\, E_{ki\leftrightarrow sk}\,(D_i)) \,\|\, HMAC_{ki\leftrightarrow sk}\,(SID_i \,\|\, D_i) \qquad (12)$$

Right after the data is *received* by S_j and *forwarded* to the next one-hop S_r, the S_j updates both pseudonyms, $HSID_{i\leftrightarrow j}$ and $HSID_{j\leftrightarrow r}$. S_j now is ready to exchange another message with S_i using the new pseudonym $HSID_{i\leftrightarrow j}$. However, S_j is not yet ready to send data to S_r since S_r does not update the $HSID_{j\leftrightarrow r}$ until D_i is forwarded to the next hop, say NS_v.

3) Acknowledgement:

As expected in data networks, a packet could get lost or corrupted. In either case, retransmission is required. Because sensors change SIDs after each transmission, synchronizing SIDs is crucial. Updating the pseudonyms depends on successful message transmission and reception. Technically, a sender and a receiver, should alter the pseudonym value only after making sure the data was sent correctly and received perfectly by the sink. The lack of direct connection between the sender and the receiver makes it a complicated process. S_i needs to calculate acknowledgement pseudonym ID ($ASID_i$) according to the expression below:

$$ASID_i = H(ASID_i \oplus \beta_i) \qquad (13)$$

The message will be sent out now with the current value for $ASID_i$. Thus, we will rewrite $M_{i\rightarrow j}$ as it appears below:

$$M_{i\rightarrow j} = \textbf{Pading} \,\|\, HSID_{i\leftrightarrow j} \,\|\, E_{ki\rightarrow j}\,(\mathbf{ASID_i} \,\|\, SID_i \,\|\, E_{ki\leftrightarrow sk}\,(D_i)) \,\|\, HMAC_{ki\leftrightarrow sk}\,(SID_i \,\|\, D_i) \qquad (14)$$

Padding is added to make sure all the one-hop packets have the same size to prevent ADV from identifying the source by *traffic analysis* and *size correlation* attacks. When S_j receives the packet, it will reformat the packet as below and then send it to S_r:

$$M_{j\rightarrow r} = \mathbf{ASID_i} \,\|\, HSID_{j\leftrightarrow r} \,\|\, E_{j\rightarrow r}(\mathbf{ASID_j} \,\|\, SID_i \,\|\, E_{ki\leftrightarrow sk}(D_i)) \,\|\, HMAC_{ki\leftrightarrow sk}(SID_i \,\|\, D_i) \quad (15)$$

The transmission of $M_{j\rightarrow r}$ should be heard by all the neighbors including both S_i and S_r. If S_i hears the message and reads *$ASID_i$*, the S_i knows that $M_{i\rightarrow j}$ was received correctly by S_j. Only at this time S_i updates the value of $HSID_{i\leftrightarrow j}$. SID_i will get updated, as well, since S_i is the source of the message. Here are two scenarios:

Scenario 1: The packet sent by S_i is lost or corrupted. In this case, S_j will not forward any message onward. Meanwhile, S_i will wait for ω time to expire. It will send the message again with updated $ASID_i$. Once the message is acknowledged according to the procedure explained earlier, and if S_i is the source, then SID_i, $HSID_i$ and $ASID_i$ will be updated. If it is intermediary sensor, only $HSID_i$ and $ASID_i$ get updated.

Scenario 2: The packet is received correctly by S_j, the new packet $M_{j\rightarrow r}$ is sent out including the acknowledgement $ASID_i$, and S_j updated the value of $HSID_{i\leftrightarrow j}$. However, S_i does not hear the forwarded message $M_{j\rightarrow r}$ within time ω. At this moment S_i does not know for sure if the packet was delivered, or the acknowledgement is lost. A copy of the message will be retransmitted to S_j with the current $HSID_i$ and updated $ASID_i$. S_j can recognize the message because of the value of old $HSID_i$. After receiving the *retransmitted* packet, it now sends direct acknowledgement to S_i as below:

$$ACK_{i\leftarrow j} = ASID_i \,\|\, Pading \quad (16)$$

Sink is treated similar to a normal sensor, so it has to acknowledge every packet it receives. After the packet is delivered to the sink, and after the message is acknowledged, the SID_i (of the source) will be updated. Both S_i and the sink will be ready to exchange a new message. As long the new message does not reach the sink before the old SID_i gets updated, the system will continue to be synchronized. This way, we have a possible window of one message. However, there is no guarantee to have all messages arrive in order or within the window time. One way to do it is if the system accounts for minimum interval time ω_{min} which is the minimum time span between two messages. However, in some cases, the packet cannot be delivered at all through a certain route. One obvious scenario is when an intermediate node gets depleted out of energy before it forwards the message onward. The message needs to be rerouted through a different intermediate node which could cause a sever delay. We suggest to implement a sliding window mechanism as exhibited in **Error! Reference source not found.**. For each sensor, we can have a window of W size. If we have k bits for the sequence number of the SID, then we can have a sliding window of *2^k-1* slots. That means the sink can receive up to W messages out of order. The window cannot slide ahead until the SIDs received earlier are all in order. We don't need to create a sliding window in the sensor side because there is no direct acknowledgement coming from the sink. Once the sink finds out that the sequence of SIDs is not in sync, it will send a message to resynchronize itself with the sensor.

4) Broadcasting data:

Typically, the sink is required to broadcast messages for control and management purposes. Likewise, a sensor might need to broadcast a message to the sink for network setup, maintenance and other management issues. A sensor could, as well, broadcast a message for emergency or urgency reasons depending on the application at hand. The framework requires keeping all the packets, transmitted throughout the network, indistinguishable. Thus, all the messages need to have similar *size*. Each sensor is preloaded with a broadcast key (kb_i) and assigned broadcast pseudonym ID ($BSID_i$). The broadcast message sent by S_i is formatted as in the expression below:

$$B = Padding \,\|\, BSID_i \,\|\, E_{kbi}(D_{bc}) \quad (17)$$

The broadcast message from a source S_i will be received by all the neighbors in N_i. S_i and the recipients will update $BSID_i$ according to the expression below:

$$BSID_i = H(BSID_i \oplus \beta_i) \quad (18)$$

Upon receiving the broadcast message, S_j decrypts the message using the stored key (kb_i) in the table T_j. It then encrypts it again using kb_j and broadcasts the message to its one-hop neighbors set (N_j) as in expression below:

$$B = BSID_i \,\|\, BSID_j \,\|\, E_{kbj}(D_{bc}) \quad (19)$$

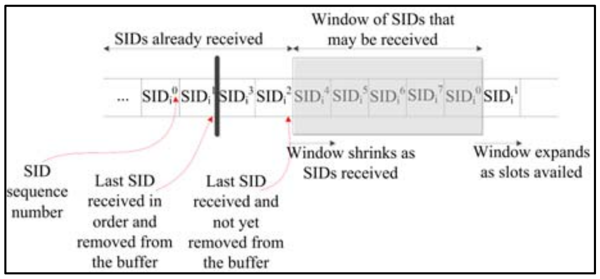
SIDs already received
Window of SIDs that may be received
...
SID_i^0
SID_i^1
SID_i^3
SID_i^2
SID_i^4
SID_i^5
SID_i^6
SID_i^7
SID_i^0
SID_i^1
SID sequence number
Last SID received in order and removed from the buffer
Last SID received and not yet removed from the buffer
Window shrinks as SIDs received
Window expands as slots availed

Fig. 1. Sliding window for received SIDs.

When the sink receives a broadcast message, it is ultimately the destination so intuitively it does not need to broadcast the message again. Our proposed framework assumes that the sink behaves like normal sensor. To maintain this requiremnt, we require the sink to broadcast the message again. Thus, we introduce the limited broadcast where the sink will broadcast only to one hop with TTL=1. Likewise, to reduce the unnecessary traffic, we have required the intermediary nodes to rebroadcast only to the neighbors with smaller *hc* in case of uplink messages, and to neighbors with bigger *hc* in case of downlink messages.

5) Broadcasting dummy packets:

The sensors need to send dummy packets to prevent *time correlation* attack and statistical analysis. The rate of the dummy packets follow a certain protocol. Dummy message is a one-hop broadcast message. However, to prevent correlation, the message needs to behave similar to real messages. So, the message needs to be encrypted and have similar size similar to other messages. This will make the dummy packets indistinguishable from the real packets. Since it has to carry a dummy data, we chose to make it carry *residual energy (Δ)* of the source sensor. This information will be extracted by the recipient neighbors and saved in the related tuple in the table T. The dummy broadcast message sent by S_i is explained in the expression below:

$$M_{dummy} = Padding \parallel DSID_i \parallel E_{kfi}(\Delta_i) \quad (20)$$

The dummy broadcast message from S_i will be received by all the neighbors in N_i. S_i and the recipients will then update $DSID_i$ according to the expression below:

$$DSID_i = H(DSID_i \oplus \gamma_i) \quad (21)$$

There is no need to worry about the pseudonym synchronization since the main purpose of dummy message is to show activity in idle sensors to obfuscate real messages.

D. Sensor cancelation and addition

If S_i needs to be canceled from WSN, it sends two messages requesting cancelation. The first message is to the sink as below:

$$M_{i\rightarrow j} = HSID_{i\leftrightarrow j} \parallel E_{ki\rightarrow j}(SID_i \parallel E_{ki\leftrightarrow bs}(D_{cancel})) \quad (22)$$

The tuple of the S_i in the sink tables will be removed. The S_i will also send a broadcast message to the neighbors according to the expression below:

$$M_{bc} = Padding \parallel BSID_i \parallel E_{kbi}(D_{cancel}) \quad (23)$$

Once the neighbors get the message D_{cancel}, they will delete the tuple related to S_i from the tables. To add S_i to the network, it will be preloaded with all setup parameters: ID_u, α_u, β_u, γ_u, H, $k_{u\leftrightarrow sk}$ and kb_u, and kdb_u. Right after deployment, S_i calculates the shared parameters with its neighbors. Before sharing the calculated parameters with the S_i, the neighbors authenticate the new sensor. The sink will send authentication key k_{add} to all sensors. S_i will be preloaded with the same key as well. S_i and the neighbors will use the key to authenticate with each other. Initially, the sink sends the following message to all of its one-hop neighbors as below:

$$B = Padding \parallel BSID_{sk} \parallel E_{kb\text{-}sk}(D_{add}) \quad (24)$$

Where D_{add} is expressed below:

$$D_{add} = hc \parallel k_{add} \quad (25)$$

The initial value for *hc* is 0 where it represents the hop count. It will be incremented every time the message is forwarded.

V. Simulation and Security Analysis

We need to analyze our solution for both passive and active adversary attacks. Sensors use pseudonyms to identify each other instead of using real IDs. The real ID's are not stored in the sensors and each pseudonym is used only once. Sensed data in the packets is encrypted all the way from the source to the destination. For *eavesdropping* and *content analysis*, ADV can intercept messages without being able to analyze them because data is encrypted. The only information the adversary can get of the captured packet is the pseudonyms: HSID, BSID or DSID which are all temporary and have no use except to calculate new pseudonyms. However, the adversary cannot get important parameters $\alpha_{i\leftrightarrow j}$, β_i or γ_i which are required to calculated new pseudonyms. For *hop-by-hop trace*, the adversary can track stream of messages from one node to another by overhearing transmitted data. The ADV will be faced with many real and dummy identical transmissions throughout the space and lifetime of the network. Furthermore, each node would retransmit through different routes and the message changes entirely after each transmission. For *size correlation*, ADV will not be able to understand relationship between incoming and outgoing packets by analyzing message sizes since all the messages have commensurate size. For *identity correlation*, ADV cannot relate overheard identities to sensors since they use different pseudonyms every time a message is transmitted. For *rate monitoring*, ADV tries to find different transmission rates in the network such as having higher transmission rate nearby the BS spatial location or at particular periods of time. This is handled by issuing dummy messages all over the network to maintain similar transmission rate chronically and spatially. If ADV compromises S_i *physically*, then it captures two sets of information: information related to the node and information related to the neighbors. The ADV would have all it needs to issue pseudonyms and send messages out to neighbors. Let's look closely at few scenarios.

Scenario 1: If the adversary physically compromises S_i, and if S_j and $S_r \in N_i$, so S_i knows some information about both S_j and S_r. However, it cannot reproduce important information such as $a_{j\leftrightarrow r}$ which is required for one-hop communication between S_j and S_r [9] because S_i would need ID_j and ID_r which are both deleted of the sensors. If S_i hears a message, it cannot determine, with high confidence, the sender among neighbors while communicating with each other. If S_i receives message from sources $\notin N_i$, then it would not be able to determine the source.

Scenario 2: If the adversary physically compromises multiple sensors, let's call it set CS, and collects number of messages, let's call it set CM. Then, the number of compromised SIDs equal to CM since each message has unique SID. If the source $S_i \notin CS$, then ADV cannot know the source sensor [13].

Scenario 3: If the message sent by source S_i as in *scenario 2* passes thought $S_j \in CS$ or even through multiple compromised sensors, it will not be able to correlate the captured SID_i to S_i.

Scenario 4: If a message sent by source S_i and $\forall S \in N_i$ is also $\in CS$ (all neighbors are compromised), then the adversary will be able to know that S_i is the source. It is unrealistic situation to have many compromised nodes in one area. However, this proves that few compromised sensors cannot locate the identity of the source. In addition, a compromised sensor does not actually need to locate the sources within its range since it can detect the objects of interest (Panda) knowing that ultimate goal is to capture the object not the sensor reporting the object.

SLP and SinkLP are achieved at first by having successful source, link and sink anonymity which was explained earlier.

The adversary cannot learn any information from the intercepted packets. Passive attacks will not endanger the location privacy. However, severe active attacks could hinder the location privacy if the WSN does not have IDS. By using dummy messages at variable interval times, it becomes hard for the adversary to correlated messages being transmitted over the network. Although shortest path routing is used in this framework, selecting the next hop is done according to a probabilistic algorithm which accounts for the residual energy levels and usage frequency to increase the routing privacy. The adversary cannot relate routes to sensors. Even if two messages follow the same route, the adversary will see them as if they are two different routes since each node along the route has different SIDs. The location safety period which is the period between capturing the first packet and finding out the location of the source or the sink. It is measured as the ratio between the period of discovering the location and the interval period [25]. We have implemented a WSN of 300 sensor nodes uniformly distributed over 200 x 200 area. Fig. 2 shows that E^2AC provides better source safety period. Fig. 3 shows that E^2AC also provides better sink safety period. However, the source safety period is much more than the sink safety period in both schemes. It is always harder to achieve SinkLP due to the volume of transmissions nearby the sink.

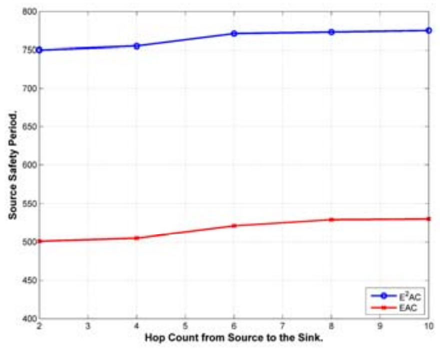

Fig. 2. Source Safety Period

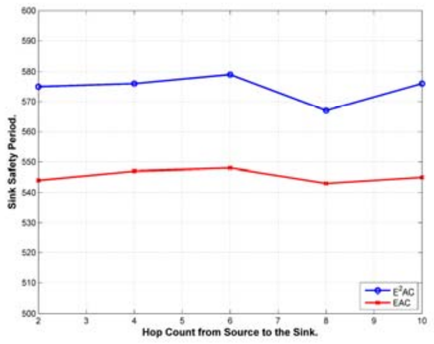

Fig. 3. Sink Safety Period

VI. CONCLUSIONS AND FUTURE WORK

Our solution provides source, link and sink anonymity, SLP and SinkLP. This work addressed local and global adversary network view. It handles both passive and active attack models. Anonymity cannot provide temporal privacy. To provide temporal privacy, dummy messages were introduced. We showed how our solution improved safety period for both SLP and SinkLP. The future work would include enhancement on the probabilistic scheme for dummy messages usage. It also will account for using dummy packets to provide efficient power consumption.

REFERENCES

[1] M. Conti, J. Willemsen, and B. Crispo, "Providing Source Location Privacy in Wireless Sensor Networks: A Survey," *Communications Surveys & Tutorials, IEEE,* vol. 15, no. 3, pp. 1238-1280, 2013.

[2] A. A. Nezhad, A. Miri, and D. Makrakis, "Location privacy and anonymity preserving routing for wireless sensor networks," *Computer Networks,* vol. 52, no. 18, pp. 3433-3452, 2008.

[3] Q. Jing, A. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the Internet of Things: perspectives and challenges," *Wireless Networks,* vol. 20, no. 8, pp. 2481-2501, 2014/11/01, 2014.

[4] Z. Yan, P. Zhang, and A. V. Vasilakos, "A survey on trust management for Internet of Things," *Journal of Network and Computer Applications,* vol. 42, no. 0, pp. 120-134, 6//, 2014.

[5] L. Yao, L. Kang, P. Shang, and G. Wu, "Protecting the sink location privacy in wireless sensor networks," *Personal and Ubiquitous Computing,* vol. 17, no. 5, pp. 883-893, 2013/06/01, 2013.

[6] S. M. a. G. Xue, "Efficient anonymity schemes for clustered wireless sensor networks," *Int. J. Sensor Networks,* vol. 1, 2006.

[7] O. Yi, L. Zhengyi, X. Yurong, N. Triandopoulos, Z. Sheng, J. Ford, and F. Makedon, "Providing Anonymity in Wireless Sensor Networks." pp. 145-148.

[8] S. Jang-Ping, J. Jehm-Ruey, and T. Ching, "Anonymous Path Routing in Wireless Sensor Networks." pp. 2728-2734.

[9] J. Chen, X. Du, and B. Fang, "An efficient anonymous communication protocol for wireless sensor networks," *Wireless Communications and Mobile Computing,* vol. 12, no. 14, pp. 1302-1312, 2012.

[10] H. Song, L. Xie, S. Zhu, and G. Cao, "Sensor node compromise detection: the location perspective," in Proceedings of the 2007 international conference on Wireless communications and mobile computing, Honolulu, Hawaii, USA, 2007, pp. 242-247.

[11] S. Zhu, S. Setia, and S. Jajodia, "LEAP+: Efficient security mechanisms for large-scale distributed sensor networks," *ACM Trans. Sen. Netw.,* vol. 2, no. 4, pp. 500-528, 2006.

[12] C. Juan, Z. Hongli, F. Binxing, D. Xiaojiang, Y. Lihua, and Y. Xiangzhan, "Towards Efficient Anonymous Communications in Sensor Networks." pp. 1-5.

[13] J. Chen, X. Du, and B. Fang, "An efficient anonymous communication protocol for wireless sensor networks," *Wirel. Commun. Mob. Comput.,* vol. 12, no. 14, pp. 1302-1312, 2012.