# MASA: End-to-End Data Security in Sensor Networks Using a Mix of Asymmetric and Symmetric Approaches

Hani Alzaid
Information Security Institute
Queensland University of Technology
halzaid@isi.qut.edu.au

Manal Alfaraj
School of Information and Communication
Griffith University
manal.alfaraj@student.griffith.edu.au

## Abstract

*Wireless Sensor Networks (WSNs) are a new technology that is expected to be used increasingly in the near future due to its cheap cost and data processing ability. However, securing WSNs is a complicated issue since the chosen security scheme should offer data security requirements such as data integrity, confidentiality, authentication, and availability although of the existed resource constraints in the sensors. In this paper, we describe the design of securing end-to-end data in WSNs using a Mixture of Asymmetric and Symmetric Approaches (*MASA*) that improves the security level offered by similar schemes and reduces the resources consumptions.*

## 1. Introduction

A Wireless Sensor Network (WSN) is composed of tiny sensor nodes which is capable of sensing some physical phenomenon, processing sensed data and communicating with each other to form an ad-hoc network capable of reporting the phenomenon to a data collection sink. Recently, WSNs have been used in many promising applications including habitat monitoring, target tracking, and battlefield surveillance. Security is crucial for majority of these applications, as sensors are typically deployed in uncontrolled and often hostile environments.

One of the potential vulnerabilities in WSNs is the security compromise of nodes, which can be achieved with relative ease, given the lack of tamper resistance packaging. An adversary can gain control of one or more nodes and readily access sensitive information such as keys, passwords. The adversary therefore can easily get access to the plain text of the encrypted messages that are routed through the controlled nodes which compromises the data confidentially. The adversary may also inject their own commodity nodes into the network by fooling nodes to believe that they are legitimate members of the network. By means of these intruder nodes (or even using compromised nodes), the adversary can inject fabricated data such as false notification of events, which can prove to be disastrous for mission-critical applications. Such attacks can also lead to exhaustion of the limited resources that are available WSNs such as bandwidth and battery power.

Moreover, WSNs are inherently resource constrained with limited energy lifetime, slow computation, small memory, and limited communication capabilities. Thus, devising security protocols for WSN is not trivial and in particular may not be successfully accomplished by simple adaptation of security solutions designed for wired networks.

Over the past few years, there has been considerable research devoted toward developing security systems for WSN. Broadly speaking, there are two different beliefs about what is the most practical scheme that is suitable for securing WSNs. On one hand, pair-wise symmetric keys are used for establishing secure channels between sensors as in [2, 5]. These schemes mainly differ in the mechanisms used for generating and distributing the shared symmetric keys. On the other hand, recent work has demonstrated the feasibility of public key cryptography in WSN as in [4, 7]. It has been shown that by employing appropriate public key algorithms and low power techniques, the energy consumption in a commodity sensor can be less than 20 $\mu$W. Almost all of these schemes however employ per-hop security semantics, wherein data confidentially and integrity is provided on a per-link basis. The lack of end-to-end security makes WSNs more vulnerable to the aforementioned attacks.

In this paper, we propose MASA that is a security system with a combination of symmetric and asymmetric cryptography to provide end-to-end data security for WSNs. MASA is based on the concept of *virtual geographic grid* wherein the entire terrain is broken down into smaller regions called *cells*. Each sensor carries two types of keys, asymmetric and symmetric. It uses the private key to sign a hashed

event notification to provide end-to-end confidentiality, authenticity, and data integrity. The symmetric key is used to authenticate the event notification within its cell and hence provide hop-by-hop authentication. Furthermore, each node maintains a list of trusted neighbors which is then used to determine the next hop node. Malicious nodes are weaned out from this list and thus ensuring data availability.

The rest of this paper is organized as follows. Section 2 provides an overview of the state of the art in WSNs security. Section 3 explains MASA in details. Section 4 summaries the data security requirements and briefly shows how MASA offers them. Section 5 concludes the paper.

## 2 Related Work

As part of the SPINS project [5], Perrig et al. proposed two security protocols, Secure Network Encryption Protocol (SNEP) and Micro Timed Efficient Stream Loss-tolerant Authentication ($\mu$TESLA) for WSNs. These protocols ensure authenticated broadcast, confidentiality/integrity for point-point transmission and data freshness. SPINS requires a shared symmetric key between each sensor and the sink. Two sensors therefore can not share a secret key between each other directly. They should share it through a trusted third party such as the sink.

Du et al. proposed a key distribution scheme that defines a threshold property to compute the probability that certain nodes are compromised [3]. For example, if the number of compromised nodes is less than the threshold, the probability that any node (except those that have been compromised) has been affected is almost zero. The threshold scheme makes the network more resilient to attacks since the attacker has to compromise a larger proportion of the nodes to create the desired impact.

Moreover, a compromised node can readily inject falsified reports into the network leading to false alarms and energy depletion. Several schemes such as [10] have been proposed to counter the node compromise attack. The general principle employed is to require $\mathcal{T}$ nodes to concur the occurrence of an event prior to relaying this event to the sink. Event notifications that are not endorsed by $\mathcal{T}$ nodes are not forwarded to the sink. However, if the attacker is able to compromise $\mathcal{T}$ nodes, it can create fake notifications with the appropriate number of endorsements.

Yang et al. proposed Location-Based Resilient Secrecy (LBRS) which overcomes this problem by adopting a location based key binding mechanism [1]. Even if $\mathcal{T}$ nodes are compromised, the attacker can only generate forged reports that claim the existence of false events in certain areas of the network without being detected. However, in all of the above schemes, a single compromised node can prevent a legitimate event report from being sent to the sink by simply offering a wrong MAC.

Ren et al. recently proposed Location-aware End-to-end Data Security (LEDS) which overcomes the aforementioned security issues by limiting the impact of compromised nodes on their vicinity [6]. LEDS divides the target terrain into several cells, known as virtual grids. A location-aware key management scheme is employed wherein each node creates the cell key by hashing its cell's location with the preloaded master key. This ensures that the effect of a compromised node is confined to its local cell. Each event report is endorsed by multiple nodes and is encrypted with a unique secret key shared between the sensing nodes and the sink. Consequently, even if nodes (other than the ones detecting the event) are compromised, the event report is not compromised. LEDS employs link-layer broadcasting to propagate an event message toward the sink. As a result, a large number of nodes within the subsequent need to participate and process the broadcasted data to ensure the data security requirements which leads to excessive consumption of resources. For example, if the number of endorsements required to confirm the validity of an event is $\mathcal{T}$ and if the detected event travels through $\mathcal{P}$ cells before reaching the sink, then a minimum of $\mathcal{T}X\mathcal{P}$ nodes must process this event. A second factor that increases the resource consumption is the use of hop-by-hop authentication. In LEDS, each event is decrypted, processed and encrypted again at each hop which leads to increased computation overhead at each hop in the forwarding path.

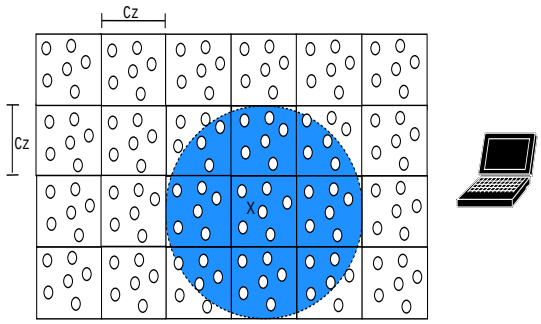


**Figure 1:** Radio Range for node $x$

## 3 The Proposed Scheme - MASA

**Assumptions.** All security systems must rely on some specific assumptions to guarantee their effectiveness. We assume that every sensor has its unique id and that they are all off-the-shelf low-cost devices without any tamper-resistant properties. We assume that the presence of a large deployment area, the dimensions of which are known in advance and that sensors are uniformly distributed over this area. The target terrain is divided into smaller cells with the dimension of each cell being small enough to allow the radio range of each sensor to cover its surrounding cells (see Figure 1). Moreover, each sensor has the ability to deter-

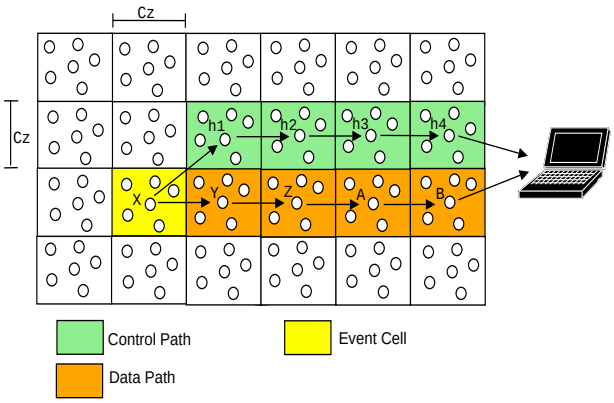


**Figure 2:** First scenario: MASA with no attack

mine its location via one of several existing localization schemes. Following the deployment of the sensor nodes, we assume the existence of short period of time (Bootstrap Phase) when the network is not vulnerable to any attacks. During this time, each node discovers its neighboring nodes and computes its cell key (explained a little later). Furthermore, each sensor is assumed to have its own private key with only the sink being aware of the corresponding public key. We assume that an event can be detected in one cell only. All detected events are collected by the sink that has the capability to do powerful computation and has sufficient memory to store all public keys.

**Adversarial Model.** This paper focuses on the detection of attacks that might target the data movement to the sink and try to eliminate the infected nodes from the trusted list. Given that the number of sensors in each cell is $n$, we assume that the adversary is capable of compromising $\mathcal{W}$ nodes where $\mathcal{W} \ggg n$ but there is no more than $R-1$ compromised nodes in any cell. When the adversary compromises a node $x$, it is able to read all $x$'s internal memory and then it could manipulate $x$ to alter the received content or even drop it. The adversary however can not compromise the sink which is secured and under the supervision of a network administrator.

### 3.1 End-to-End Data Security

Each node is preloaded with the following parameters

$$\{\mathcal{K}_m, \mathcal{K}_p, \mathcal{C}_z, \mathcal{R}, \mathcal{B}, t\}$$

where $\mathcal{K}_m$ denotes a master key, $\mathcal{K}_p$ denotes the node's private key, $\mathcal{C}_z$ denotes the cell dimension (we assume square cells), $\mathcal{R}$ denotes required confirmation messages that should be received from cell members to validate an event, $\mathcal{B}$ denotes the location of the sink, and $t$ denotes the threshold time that any node waits until it gets a response from the subsequent node along the path to the sink. For simplicity, MASA is divided into the following four phases:

**Bootstrap Phase.** Each sensor computes the center of its cell's, $l$, by using an existing localization scheme. Each sensor then computes the cell key $\mathcal{K}_c$, which is used to authenticate any communication within the cell, as follows:

$$\mathcal{K}_c = \mathcal{H}(\mathcal{K}_m|l)$$

Following this, each node broadcasts its existence to its neighbors within the transmission range. To avoid collisions, each sensor senses the channel and uses techniques such as back-off prior to broadcasting [1]. On receiving such broadcasts from its neighbors, each node populates a list of trusted neighbors which consists of node $ids$ and locations. At the end of this phase, each node deletes its master key to prevent an adversary from deriving the keying material of other nodes.

**Generation Phase.** The generation phase oversees the operations that control the detection of an event and its ratification followed by the initial transmission of the event message to the next-hop node from the event source. Once a node $x$ (see Figure 2) detects an event, it shares this knowledge with its cell members. A confirmation message is created consisting of

$$\{\mathcal{S}, \mathcal{T}, TimeStamp, EventType\}$$

where $\mathcal{S}$ denotes the sender of the event and $\mathcal{T}$ denotes the packet's type (includes event, confirmation message, choosing a helper node, etc). The node encrypts this confirmation message by $\mathcal{K}_c$ and broadcasts it. To consider a detected event as a real legitimate event, a node should receive $R$ confirmation messages from distinct cell members that present in its trusted list. On the contrary, if less than $\mathcal{R}$ confirmations are received, then the originator of the original event will be labeled as malicious by other nodes and removed from the trusted list. The value of $\mathcal{R}$ is determined by the network administrator and preloaded into each sensor prior to deployment. The choice of the value of $\mathcal{R}$ represents a trade off between the level of security offered and the likelihood of false positives. A large value of $\mathcal{R}$ implies that an adversary would need to compromise a large number of nodes inside a cell to fake an event. However, this increases the chance of false accusations against well-behaved nodes since the event may have been genuinely detected by less than $\mathcal{R}$ nodes in the cell.

Once the number of received confirmation messages is $\geq \mathcal{R}$, then $x$ generates an event message to be sent to the sink as follows:

$$\{\mathcal{S}, \mathcal{T}, TimeStamp, \mathcal{SI}, \mathcal{MD}\}$$

where $\mathcal{SI}$ denotes Sensitive Information that a node might include in the event message. $\mathcal{MD}$ denotes the message digest of the event. The event is digitally signed by applying one-way hash function to it and then encrypted with the

private key of $x$. In addition to this signed part, the event message also contains the source id ($\mathcal{S}$), and event type ($\mathcal{T}$) in plain text. Consequently, the intermediate nodes that route this message toward the sink do not need to decrypt the content of the message in order to perform the forwarding function which in turn reduces the power consumption at these nodes. This is in contrast to [6] where each relay node needs to first decrypt the message prior to forwarding. $x$ then seeks to establish two paths toward the sink: one for forwarding the event message (a.k.a data path) and the other for monitoring the progress of the event message from one cell to the next (a.k.a control path). Nodes along the control path are referred to as *helper nodes*. Any existing routing protocols such as AODV could be used for setting up these paths. We however choose to rely on the class of geographic routing protocols since it is easier to integrate them with the cell structure employed in MASA.

The next hop nodes chosen by $x$ (along the data and control path) must satisfy the following conditions: they must be from its trusted neighbor list, they must be closer to the sink by $\mathcal{C}_z$, i.e. one cell closer to the sink, and they must be within each others communication range. Figure 2 explains an example where $x$ has chosen $y$ as next hop along the data path and $h_1$ as the corresponding helper node. The function of the helper node is two-fold. Firstly, it must oversee the selection of the next hope node chosen by $y$ and ensure that it is one cell closer to the sink which prevents the possibility of wormhole attacks [9]. Secondly, the helper node must overhear the subsequent transmission by the forwarding node along the data path and ensure that the message has not been altered or dropped which ensures message integrity and data availability. Once $x$ selects $h_1$, it unicasts a control message asking it to monitor the movement of the event message from $y$ to the next cell as follows:

$$\{\mathcal{S}, \mathcal{T}, \mathcal{D}, TimeStamp\}$$

where $\mathcal{D}$ denotes the node that the helper node should monitor which is $y$. $x$ then starts a timer set for some duration $t$ and unicasts the event message to $y$. It then switches to promiscuous mode and waits for $y$ to forward the event message to a node in its neighboring cell (node $z$). If the timer expires and $x$ did not hear the relay transmission from $y$, $y$ is removed from $x$'s trusted list of neighbors. At the same time, $x$ broadcasts a message to its neighbors informing them about the malicious behavior of $y$.

Suppose the location of $x$ is $(x_1, y_1)$, and the location of $z$ is $(x_3, y_3)$. Node $x$ ensures that node $y$ forwards the event packet within a certain time $t$ to $z$ such that,

$$2\mathcal{C}_z \leq \mathcal{D}_l \leq 3\mathcal{C}_z. \quad (1)$$

where $\mathcal{D}_l = \sqrt{|x_3 - x_1|^2 + |y_3 - y_1|^2}$.

This ensures that node $z$ is closer than node $x$ to the sink by $2\mathcal{C}_z \sim 3\mathcal{C}_z$. If $z$ is chosen such that it does not satisfy Eq.(1), then a broadcast message is sent by $x$ identifying $y$ as a malicious node. $x$ then picks an alternate node as a replacement of $y$.

**Forwarding Phase** The forwarding phase includes the hop-by-hop forwarding of the event message until it reaches the sink. $y$ repeats what $x$ did in the generation phase except that it does not choose the second helper node $h_2$ which will be chosen by $h_1$. $z$ is out of $x's$ radio range and thus $x$ uses $h_1$ to monitor the event movement one cell closer toward the sink (see Figure 2). $h_1$ also should be in the radio range of $z$. It computes

$$\mathcal{D}_l = \sqrt{|x_4 - x_2|^2 + |y_4 - y_2|^2}$$

where $(x_4, y_4)$ is the location of $a$ as a destination of node $z$ and $(x_2, y_2)$ is the location of $y$ (see Figure 2). The main function of $h_1$ is to ensure that the chosen $a$ is in the range of the Eq.(1). Once $h_1$ notices that the chosen destination is not in the range of Eq.(1), it considers $z$ as a malicious node and broadcasts this to its neighbors to update their trusted list. $y$ then chooses another node instead of $z$ to be its new destination. The event packet is forwarded by the intermediate nodes till it reaches the sink.

**Verification Phase** An event is verified at the sink by the signature of the node that created the event. The sink can verify whether the received event is sent by a specific node or not as follows: it calculate the hash code of the event, decrypt the received message digest MD, and then compare the two message digests. If the calculated hash code does not match the result of the decrypted signature, either the document was changed after signing or the signature was not generated with the private key of the generator node. Let us consider another scenario with a compromised node $a$ (see Figure 3). In this case, the packet will be forwarded till it reaches $a$. $h_2$ is listening to $a$ to figure out who is the $a's$ destinations to choose a proper helper node. However, $a$ drops the packet instead of forwarding it to the correct destination. After $t$ time, $z$ will notice that and re-choose another destination and forward the packet to it. $h_2$ chooses the next helper node that closer to the sink by $\mathcal{C}_z$ and in the radio range of $m$. $m$ then forwards the packet to $n$ and finally to the sink.

## 4 Security Requirements

The data security requirements in WSN are similar to those in traditional network [8]. In this section, we formalize the required security properties and explain how MASA achieves these properties.

**Data Integrity.** It ensures that the content of the message has not been altered, either maliciously or accidentally, in transmission. MASA ensures data integrity as follows: it signs the sensitive information by the originating node's private key and only the sink has the corresponding public

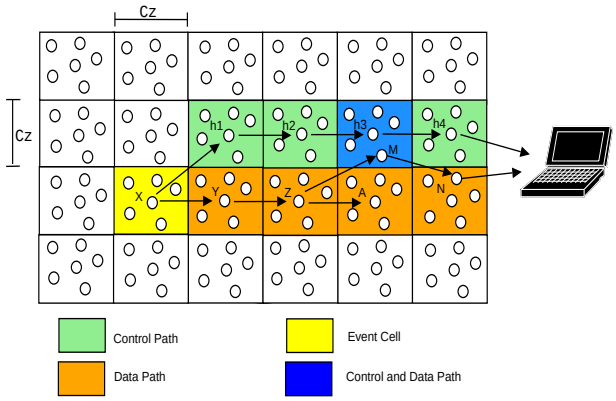


**Figure 3:** Second scenario: MASA with an attack

key. The sink thus can verify that no one has tampered with this sensitive data. Secondly, hop-by-hop data integrity is achieved by creating the control path to monitor the event movement.

**Data Confidentiality.** It ensures that only the sender and intended receiver should be able to understand the content of the transmitted message. The intermediate nodes should route messages with no access to the sensitive part of the content. In MASA, the valuable part of the event is signed by a private key and no one other the sink has access to it. However, there is unencrypted part of the event that allows the intermediate nodes to route the event with no computation overhead.

**Data Authentication** It allows a receiver to verify whether the message is sent by a claimed sender or not. They might share a secret key to compute a message authentication code MAC of all the transmitted data. However, it is been discussed in [5] that using a symmetric MAC to send authentic data to untrusted receiver is insecure unless making a strong assumptions. In MASA, we use a list of trusted neighbors and helper nodes to control the data movement between a node and the sink.

**Data Availability.** Data availability ensures that the network is alive and the data is accessible. It is highly recommended in the presence of compromised nodes to achieve network degradation by eliminating these nodes. In MASA, each node keeps a list of trusted neighbors and once a node behaves maliciously, it will be removed from the list. This leads to reduce the number of trusted nodes in the networks or degrade the network.

## 5 Conclusion

By reviewing the existing location-aware end-to-end data security in WSNs such as LEDS, we came up with MASA that improves some of the weaknesses in LEDS such as using broadcast to sent an event to the sink, and doing complicated functions at each hop in the forwarding path i.e decrypt the old MAC, and compute new MAC and encrypt it again. In MASA, all the decryption operations are shifted to the sink side since it has enough resources.

## References

[1] Nael Abu-Ghazaleh, Kyoung-Don Kang, and Ke Liu. Towards resilient geographic routing in wsns. In *Q2SWinet '05: Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*, pages 71–78, New York, NY, USA, 2005. ACM Press.

[2] Haowen Chan, Adrian Perrig, and Dawn Song. Random key predistribution schemes for sensor networks. In *SP '03: Proceedings of the 2003 IEEE Symposium on Security and Privacy*, page 197, Washington, DC, USA, 2003. IEEE Computer Society.

[3] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney. A key management scheme for wireless sensor networks using deployment knowledge, 2003.

[4] J. Kaps G. Gaubatz and B. Sunar. Public keys cryptography in sensor networks. In *The Proceedings of the 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS)*, 2004.

[5] Adrian Perrig, Robert Szewczyk, Victor Wen, David E. Culler, and J. D. Tygar. SPINS: security protocols for sensor netowrks. In *Mobile Computing and Networking*, pages 189–199, 2001.

[6] Kui Ren, Wenjing Lou, and Yanchao Zhang. LEDS: Providing location-aware end-to-end data security in wireless sensor networks. *IEEE Trans. Mob. Comput.*, 7(5):585–598, 2008.

[7] Rodrigo Roman and Cristina Alcaraz. Applicability of public key infrastructures in wireless sensor networks. In Javier Lopez, Pierangela Samarati, and Josep L. Ferrer, editors, *EuroPKI*, volume 4582 of *Lecture Notes in Computer Science*, pages 313–320. Springer, 2007.

[8] Elaine Shi and A.perrig. Designing secure sensor networks. *Wireless Communication Magazine*, page 11(6), 2004.

[9] A. Perrig Yi-Chun Hu and D. B. Johnson. Wormhole attacks in wireless networks. *IEEE Transactions on Selected Areas in Communications*, 24:370–380, Feb 2006.

[10] S. Zhu, S. Setia, S. Jajodia, and P. Ning. An interleaved hop-byhop authentication scheme for filtering false data injection in sensor networks, 2004.