Time-slotted voting mechanism for fusion data assurance in wireless sensor networks under stealthy attacks

By: Hung-Ta Pai, Jing Deng, Yunghsiang S. Han

H.-T. Pai, J. Deng, and Y. S. Han (2010), "Time-Slotted Voting Mechanism for Fusion Data Assurance in Wireless Sensor Networks under Stealthy Attacks," *Computer Communications*, vol. 33, no. 13, pp. 1524-1530.

Abstract:

In wireless sensor networks, data fusion is often performed in order to reduce the overall message transmission from the sensors toward the base station. We investigate the problem of data fusion assurance in multi-level data fusion or transmission in this paper. Different to a recent approach of direct voting where the base station polls other nodes directly regarding to the received fusion result, we propose a scheme that uses the time-slotted voting technique. In this scheme, each fusion node broadcasts its fusion data or "vote" during its randomly assigned time slot. Only the fusion result with enough number of votes will be accepted. Thus, our scheme eliminates the polling process and eases the energy consumption burden on the base station or the fusion data receiver, which could well be the intermediate nodes. Our analysis and simulation results support our claim of superiority of the proposed scheme.

Keywords: Wireless sensor networks, Data fusion, Data aggregation, Information assurance, Stealthy attack

Article:

1. Introduction

Wireless sensor networks (WSNs) comprise many tiny, low-cost, battery-powered sensors in an area of interest [1,2]. The sensors detect environmental variations and then transmit the detection results to a base station [3,4]. The detection results are usually processed before they are transmitted to the base station, reducing the overall transmission energy consumption. This process is called *data fusion* [5–7] or *data aggregation* [8,9]. The sensor that collects the detection results from other sensors and performs the data fusion is a *fusion node*. The fusion data are then sent to the base station through multiple hops or a direct link (i.e., one hop).

Although fusion significantly lowers the traffic between the fusion node and the base station, the fusion node is more critical and vulnerable to malicious attacks than regular sensors [10,11]. If a fusion node is compromised, then the base station cannot ensure the correctness of the collected fusion data. This problem of fusion data assurance arises because the detection results are not sent to the base station at all and the fusion result cannot be verified. In WSNs with multi-level of data fusions or transmission, this problem becomes more severe as the early fusion results are fused multiple times before reaching the base station and the intermediate fusion nodes may be compromised.

Several approaches have been proposed to address this attack in WSNs with fusion results directly transmitted to the base station [12–15]. Some of the approaches may be easily extended to a multi-hop version [13]. Some works will be described in detail in next section. The scheme by Deng et al. [16] is specifically designed to assure the fusion data received at the base station in the multi- hop network. The method constructs multiple paths between the fusion node and the base station. The base station obtains a copy of the fusion result through each route and then compares all of the received fusion results. However, as the potential number of compromised nodes increases, the number of required paths increases. Furthermore, transmitting several copies

of the fusion results toward the base station consumes a large mount of power. Therefore, an efficient method for fusion data assurance is needed.

This work proposes a time-slotted voting scheme combining the concept of time division multiple access with direct voting mechanism proposed by Pai and Han [15]. In the direct voting mechanism, the base station receives the fusion data from a randomly chosen fusion node and then polls other sensor nodes, as witnesses, for votes on the fusion data. The vote is exploited to verify the correctness of the fusion data and the base station consumes some extra power to verify the data fusion results. In WSNs with multi-level of data fusion or transmission, however, the intermediate fusion nodes do not have such extra power to verify the incoming fusion results. Therefore, the direct voting mechanism will not work well in WSNs with multi-level of data fusions or transmission. Instead, we propose to use a proactive verification technique. In this technique, each fusion node is assigned a time slot to broadcast its fusion result, eliminating the polling process completely. The key advantages of the proposed scheme are summarized as follows:

- All properties of the direct voting scheme proposed in [15] are still present: low transmission, short verification delay, exclusion of compromised nodes, no extra memory requirement, and reject of forged result when the number of compromised nodes is less than the number of required support votes.
- Every data fusion node has the same task and the assurance process in each hop is performed cooperatively by a group of sensors. Thus, data fusion assurance can be implemented in the WSN easily.
- No iterative process is needed. The assurance process can be performed with the transmission of the fusion result to the base station. Therefore, only a fixed and short delay is added to the process.

The remainder of this work is organized as follows. Section 2 briefly addresses the problem of data fusion assurance in WSNs and previous works on the problem. Section 3 describes the proposed scheme and its performance and overhead. Section 4 presents a performance evaluation of the proposed approach. Concluding remarks and suggestions for future work are presented in Section 5.

2. Data fusion assurance problem and the previous work

Fig.1 depicts a WSN for distributed detection with N sensors for collecting environment variation data. The collected data are transmitted to a fusion node from all of the sensors. The fusion node yields a final result according to the data, and sends the final result to a base station directly.

Two problems must be addressed to ensure that the base station obtains the correct result. First, the fusion node must correctly fusion all of the collected data. Several algorithms have been proposed to deal with this problem [4,17]. The second problem concerns assurance of the fusion result (transmission between the fusion node and the base station is assumed herein to be error- free). Since the fusion node may be compromised, forged data may be transmitted to the base station, which has no way to detect such forged results. This is the so-called *stealthy attacks*, where an attacker tries to trick the base station to accept a forged result [12]. This work only focuses on the stealthy attack but not others. The main idea of the proposed scheme is to avoid using integrity mechanism such as Message Authentication Codes (MACs) which intro duce extra transmission overhead. Since only stealthy attack is considered, for which assurance of receiving correct fusion data is the main issue, a simple voting scheme is proposed. The fusion result is broadcast such that compromised node can only jam the signal but cannot modify it. However, we do not consider denial-of-service attack in this work. The only overhead that is taken into account is the extra effort for preventing the network from the stealthy attack.

Chan et al. [12] developed an assurance process for data fusion using a Merkle tree. The fusion node establishes a Merkle hash tree using collected detection results as leaves. The base station requests one of the results and checks if it is consistent with the tree during the assurance process. The probability of detecting a cheating fusion node can be increased by transmitting more detection results to the base station. However, individual assurance algorithm must be developed for each of different fusion operations such as average, sum, and maximum. General one-for-all assurance approach is difficult to design. Additionally, only one fusion node is assumed to be able to communicate with the base station. When it is compromised, the base station can no

longer receive correct fusion data. This method was later extended to the case where the fusion is performed over a fusion tree including multiple fusion nodes as internal nodes [13]. In [13], MACs are applied to the assurance process and the technique can detect multiple compromised nodes. Nevertheless, when any fusion node is compromised or faulty, the assurance process fails and the network stops functioning. Moreover, the communication loading for the assurance process is $O(M\Delta H^2K)$, where M is the number of nodes in the network, Δ is the degree of the fusion tree, H is the height of the fusion tree, and K is the length of the MAC. Frikken and Dougherty further reduced the communication loading to $O(M\Delta HK)$ [18].

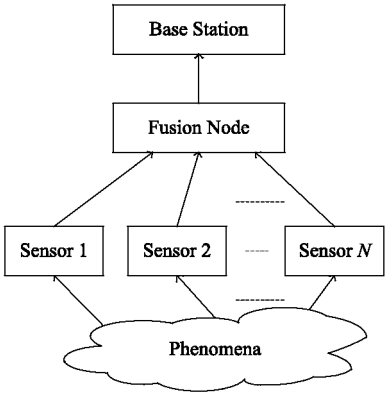

Fig. 1. Structure of a wireless sensor network for distributed detection using N sensors and one fusion node.

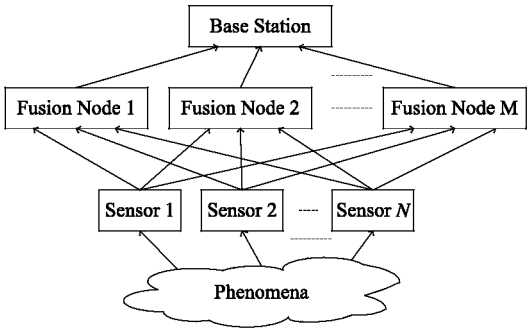

Fig. 2. Structure of a direct-linked wireless sensor network for distributed detection using N sensors and M fusion nodes.

Du et al. [14] developed a witness-based approach to ensure the correctness of the fusion result as illustrated in Fig. 2. Pai and Han [15] proposed a direct-voting mechanism to improve the witness- based approach. The base station chooses a fusion node to transmit the fusion result and then polls other fusion nodes one by one to check whether they agree with the transmitted fusion result. The base station obtains the vote from the witness node directly instead of through the chosen node. No forgery problem exists and only one correct fusion result will be transmitted to the base station. However, the base station consumes more energy to conduct the assurance process because several rounds of polling may be necessary to obtain a valid fusion result. Moreover, when multiple hops are needed to transmit the fusion result to the base station, this scheme cannot be employed since the intermediate fusion nodes between the base station and the fusion node may be compromised. Thus, when data fusion results are sent through multiple hops, fusion assurance should be provided on the intermediate nodes, which cannot simply take the base station's role because their limited battery power.

3. Time-slotted voting mechanism

3.1. Network structure and algorithm

The structure in Fig. 2 is extended to a multi-hop fusion scenario. Fig. 3 illustrates the network structure for data fusion assurance in such WSNs with N sensors and M_1 fusion nodes. H hops are required to transmit the fusion result to the base station from the fusion node. At the hth hop, $h = 1,2,...,H— 1$, M_{h+1} fusion nodes[1] are grouped at the $h + 1$ layer to receive and forward the fusion result. Note that the relation between M_{h+1} and M_h is arbitrary. Local time synchronization is assumed at each layer.[2] The base station obtains the fusion result at the Hth hop (the final hop). This network structure can be found in clustered WSNs [20]. Several real multi- hop sensor networks can be found in [21].

In the direct-voting scheme, the base station consumes most power in the polling process. In this work, we propose a time-slotted voting approach, in which every fusion node at one layer of the multi-hop WSN transmits its vote or fusion result to the fusion node at the next layer in a pre-assigned and fixed time slot. With such pre-assigned time-slotted transmissions, no polling is necessary.

We use the fusion process of the hth layer fusion nodes as an example for discussion. Fusion nodes of other layers follow the same procedure. Assume there are M_h fusion nodes at the hth layer.

In general, each of the M_h fusion nodes gets a chance to submit its data or vote, unless it is unnecessary (this point will become clear later). The transmission schedule for these M_h fusion nodes can be rotated to balance their power consumption. All fusion nodes will listen while other fusion nodes at the same layer transmit. A threshold T_h is used in order to decide whether a certain result has obtained enough votes.

Without loss of generality, we name the M_h fusion nodes at the hth layer as node 1,2,...,i,...,M_h according to the sequence of their transmission schedule. Therefore, node *1* sends first and node 2 sends next, and so on. When it is node i's turn to transmit, it will choose the first clause that agrees with its observation thus far:

C1. If more than T_h votes have been submitted to support a certain fusion result, node i remains silent.
C2. If no fusion result has received at least T_h — (M_h — i) votes, node i remains silent.
C3. If there has been a fusion result transmitted earlier on (by one fusion node whose transmission schedule is ahead that of node i), node i will send an agreement vote to support this result.
C4. Otherwise, it will send its fusion result.

At the end of the transmission time slots of all fusion nodes at this layer, if there exists a data fusion result with at least T_h supporting votes, this result will be accepted.

We explain the intuitive reasons to follow the different clauses in our algorithm as below. Our algorithm ensures that a data fusion node sends its data only when it is necessary to do so. The objective of each hop is to ensure an agreement on a correct data fusion result. If there has been such a result with more than T_h supporting votes (including the original sender), other nodes do not need to vote (hence clause C1). In some scenarios, the nodes close to the end of the transmission sequence may see that, even if all the rest of fusion nodes agree with the currently most-popular result, there is no way to come up with a result with at least T_h supporting votes. Therefore, it is useless to send in more votes or even new data (hence clause C2). Clause C3 simply states that a fusion node will send an agreement vote to support a result that has been submitted before. Clause C4 makes sure that someone will submit new results when there is a chance to obtain enough votes.

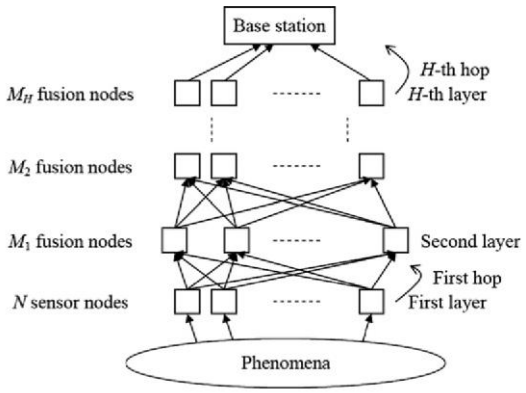

Fig. 3. Structure of a multi-hop wireless sensor network for distributed detection using N sensors, M_1 fusion nodes, and M_h fusion nodes as intermediate nodes, $h = 2,3,\ldots,H$, at the hth hop.

3.2. Discussions of the algorithm

We have the following discussions of the algorithm that we just explained:

Transmission schedule. The order of the time slots is important. The node with the first time slot, i.e., the first node, must transmit its fusion result. Assuming that, in general, the fusion result has more bits than the vote, the first node will consume more power than other nodes. Moreover, if the first node is compromised, all other nodes will have to send more bits, on an average, as shown in Section 3.3. Since no information about compromised nodes is known, the order should be random and should be changed periodically such that all nodes consume approximately the same power. This will lead to a more balanced power consumption of sensors in the WSN.

Promiscuous mode. Data broadcasting techniques are adopted in this scheme because the data must be received at all fusion nodes of all layers. All nodes must save every received fusion result in order and the number of its corresponding supports (i.e., votes). Such records play a critical role for each node, e.g., node *i*, to decide which of the clauses among C1, C2, C3, and C4 to take.
Multi-layer fusion. All fusion nodes employed in Fig. 3 have the same inputs and outputs. However, this network structure model can be extended to a network with multi-layer fusion [13]. The nodes in every layer may be divided into several groups such that a tree-structured network is formed. The nodes in a fusion group can fuse the detection results of the sensors in the same fusion group and then securely get fusion results from the fusion groups of the previous (lower) layer. Finally, the nodes can combine all of these fusion results to produce a new fusion result and transmit it to the next (higher) layer. Note that every node in the network may have the same functions, including data sensing, transmitting, data fusion, and data fusion assuring.

3.3. Performance analysis

First of all, since the assurance process is performed with the transmission of the fusion result to the base station, the delay is fixed and equals to M-τ, where τ; is the duration of one time slot, and

$$M = \sum_{h=1}^{H} M_h.$$

In order to reduce the delay, the number of fusion nodes involved in forwarding fusion results should not be too large. However, smaller number of fusion nodes at each layer results in higher probability for the base station to receive forged fusion results. Hence, there is a tradeoff between the delay and security strength.

Next, the performance analysis of the proposed scheme is divided into two parts: security and communication traffic. In the security analysis, the majority voting rule is assumed to be adopted in every hop of the scheme, where $T_h + 1 \geq [M_h/2]$ and M_h is odd. We also assume that the fusion result received by the fusion nodes at first layer is always correct. This assumption is needed for us to evaluate the effect of hop-transmissions. Moreover, we assume that all compromised nodes collaborate with each other to make the base station accept a forged fusion result. The base station accepts a forged fusion result only if the number of compromised nodes in hth layer, C_h, exceeds T_h and the compromised nodes cooperate with each other. Based on the assumption that a node may be compromised with an i.i.d. probability p_c, the probability that a forged fusion result is accepted in the hth hop is given by

$$E_h = \sum_{i=T_h+1}^{M_h} \binom{M_h}{i} p_c^i (1-p_c)^{M_h-i} \quad \text{for } 1 \leqslant h \leqslant H.$$

When more than T_h nodes are not compromised, the correct fusion result is obtained in the hth hop. Accordingly, the probability that a correct fusion result is accepted in the hth hop is represented by

$$R_h = \sum_{i=T_h+1}^{M_h} \binom{M_h}{i} (1-p_c)^i p_c^{M_h-i}.$$

If a correct fusion result is received in the base station, then in each hop, it must be received successfully. Hence, the probability that the base station receives a correct fusion result, R, is then

$$R = \prod_{h=1}^{H} R_h.$$

The probability that the base station accepts a forged result is derived for two cases. The first case is that the compromised nodes in the same hop cooperate with each other but the compromised nodes in the different hops do not cooperate (partial cooperation). Let F_h^s be the probability of the event A_h^s that a forged result is accepted at hth hop and the result is successfully transmitted to the base station. Then

$$F_h^s = E_h \prod_{i=h+1}^{H} R_i.$$

The probability that the base station accepts a forged result is the probability of the union event of A_h^s for all $1 \leq h \leq H$. It is clear that A_i^s and A_j^s are disjoint events for all $i \neq j$. Hence, the probability that the base station accepts a forged result is

$$E^s = \sum_{h=1}^{H} F_h^s.$$

The second case is that the compromised nodes in all hops cooperate with each other (full cooperation). Let F_h^a be the probability of the event A_h^a that a forged result is accepted at *h*th hop and but no forged result is accepted prior to this layer:

$$F_h^a = E_h \prod_{i=1}^{h-1} R_i.$$

The probability that the base station accepts a forged result is the sum of all chances of a forged result is accepted at each of the layers:

$$E^a = \sum_{h=1}^{H} F_h^a.$$

The calculation is due to our assumption of all compromised nodes collaborating to each other. For instance, if a forged result is accepted at *j*th layer, the compromised nodes at higher layers, $j + 1$, $j + 2$,..., *H*th layers, will make no attempt to change this result.

The above equations indicate that if T_h is higher, the forged fusion result is accepted with lower probability and the correct fusion result is rejected more easily. The choice of T_h is therefore critical to balance between rejecting forged results and rejecting correct results. Fig. 4 compares the probabilities that the base station accepts a forged fusion result and a correct result when M_1=21, M_2=17, M_3=13, M_4=11, M_5=7, T_1=10, T_2=8, T_3=6, T_4 = 5, T_5 = 3, and H = 5. For two cases, the probability that the base station accepts a correct result is much larger than the probability that the base station accepts a forged result when $p_c < 0.1$. In contrast, the probability that the base station accepts a forged result is larger than the probability that the base station accepts a correct result when $p_c > 0.34$. Moreover, it is not surprised that the case of the partial cooperation in Fig. 4(a) has a little lower probability of accepting a forged result than the case of the full cooperation in Fig. 4(b). However, the probabilities of accepting a correct result in both cases are identical. Notice that it is possible that no result is accepted.

Since the node transmits data by broadcast, the communication traffic must be divided into two parts: transmission and reception. In short range communications, the reception dominates the power consumption. Nevertheless, the transmission consumes much more power than the reception in long range communications [22]. Because the number of the receiving nodes at each transmission equals to the number of nodes at two neighboring layers, the reception power can be easily derived from the transmission traffic. Hence, we will only focus on the derivation of transmission traffic.

The analysis for the transmission traffic assumes that a compromised node always disagrees with the correct fusion result and transmits the forged fusion result. Whether such a compromised node prefers to send a completely different fusion result depends on Pf: If the compromised node attempts to trick the receive node to accept the forged fusion result, then it always agrees with the fusion result transmitted by other compromised nodes, i.e., P_f=1. If P_f= 0, however, the compromised node always disagrees with the current fusion results.

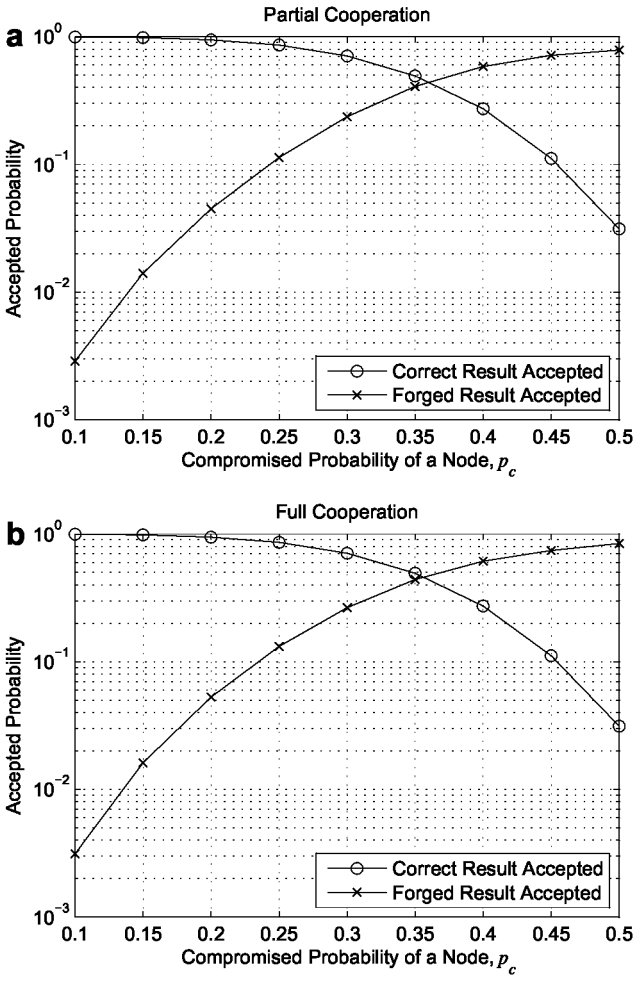

Fig. 4. Comparison of probabilities that base station accepts a forged fusion result and a correct result when $M_1 = 21$, $M_2 = 17$, $M_3 = 13$, $M_4 = 11$, $M_5 = 9$, $T_1 = 10$, $T_2 = 8$, $T_3 = 6$, $T_4 = 5$, $T_5 = 4$, and $H = 5$ (a) when only the compromised nodes in the same hop cooperate with each other, and (b) when all compromised nodes cooperate with each other.

In this work, *overhead* is adopted as a transmission traffic index. The overhead is defined as the total number of bits, except the bits for one copy of the correct fusion result, transmitted to the base station by *uncompromised* nodes during the data assurance process. Furthermore, we only consider the complexity when a correct fusion result can be sent to the base station since when a forged one is accepted by the base station or when no result is accepted by the station the network is either useless or must be re-organized or re-deployed. Since only receiving correct fusion result is considered, both attack models mentioned previously will have the same complexity.

For a transmit node, denote $k_h(v)$ as the number of bits required for agreeing with the vth transmitted fusion result. Note that $k_h(v)$ may include packet headers that are needed for transmission. If P_f=1, the correct fusion result transmitted by an uncompromised node is the first transmitted result with probability $(M_h - C_h)/M_h$ or the second transmitted result with probability C_h/M_h. When the fusion result is valid in the hth hop, i.e., the number of uncompromised nodes is greater than the threshold ($M_h - C_h > T_h$), the number of agreeing votes on the correct fusion result is T_h. The average overhead at the hth hop is then given by[3]

$$O_h(p_f = 1) = \frac{M_h - C_h}{M_h} k_h(1) T_h + \frac{C_h}{M_h} k_h(2) T_h.$$

We can find that the overhead increases as Th increase. That is the overhead is O(Th) in the hth hop. The total overheads are represented by $\sum_{h=1}^{H} O_h(1)$. Furthermore, every receive node only has to take and save two versions of fusion results. One is correct and the other is forged.

When $P_f = 0$, every receive node at the hth layer must take and save at most $C_h + 1$ versions of fusion results, where one is correct and the others are forged. The overhead at the hth hop is related to the time slot where the first uncompromised node appears to transmit its fusion result. The probability that the first uncompromised node is located at the vth node, v=1,2,...,C_h+ 1, i.e., v —1 compromised nodes transmit their fusion results before it, is represented by

$$P_0(v) = \frac{\binom{M_h - v}{C_h - v + 1}}{\binom{M_h}{C_h}}.$$

The above analysis is derived as the problem of counting for C_h black balls (compromised nodes) and M_h — C_h white balls (valid uncompromised nodes) together since all compromised nodes (uncompromised nodes) have the same behavior. Similar to the case of $P_f = 1$, when the fusion result is valid, the average overhead in the hth hop is given by

$$O_h(p_f = 0) = \sum_{v=1}^{C_h+1} P_0(v) k_h(v) T_h.$$

Similarly, the overhead is $O(T_h)$ in the hth hop and the total overhead is the summation of the overheads of all hops, i.e.,

$$\sum_{h=1}^{H} O_h(0).$$

Next we describe a method to assign $k_h(v)$ for different vs. When the transmit node agrees with the first transmitted result, it sends nothing. When the transmit node agrees with the vth transmitted result, where $v > 1$, it sends the value of v — 2. Thus,

$$k_h(v) = \begin{cases} 0 & v = 1, \\ 1 & v = 2, \\ \lceil \log_2(v-1) \rceil & v > 2. \end{cases}$$

Fig. 5 shows the overheads in the hth hop while M_h =11, Th = 5 and Mh = 21, Th = 10. The case of Pf = 0 has higher overhead than the case of Pf =1. Notice that the base station accepts the forged fusion result when $C_h > 5$ for $M_h = 11$ and $C_h > 10$ for $M_h = 21$. By Fig. 5, there are less than three and six bits of overhead in one hop for $M_h = 11$ and 21, respectively. Notice that the overhead is strongly related to the number of compromised nodes, not the fusion nodes. Thus, only the case of M_h =11 is shown in the following simulations. Similar results can be found in when $M_h = 11$, $T_h = 10$ as shown in Fig. 6. The case considering the packet size will be given in next section.

When the communication range between two layers is long, the transmission traffic dominates the power consumption. The overhead of the proposed scheme is $O(THk)$, where T is the maximum number among all T_h and k is the maximum number of bits in any agreed message which contains the bits required for agreeing. Since T is much less than the total number of nodes, M, and k is usually less than the length of MAC,[4] the power consumption of the proposed method is much lower than the method in [18], $O(M\Delta\ HK)$. On the other hand, when the communication range is short, the power consumption of the reception must be considered. The communication of the proposed scheme becomes $O(T\Lambda Hk)$, where $\Lambda = \max\{M_1, M_2, ..., M_H\}$, because the number of receiving nodes at the h-hop is M_h—1 + M_h. Since $\Lambda \approx \Delta$, the power consumption of the proposed scheme is still lower than the method in [18].

3.4. Non -fully connected networks

The algorithm detailed in Section 3.1 requires that all sensor nodes in the same data fusion group can hear each other. When these nodes are not fully connected or when collisions or packet loss occur at the overhearing nodes, it is possible that some sensors cannot hear from others. Based on our algorithm (Step 2d), such nodes

may send data fusion results instead of their votes. Although this increases the voting overhead, which depends on the chance of nodes not overhearing from other nodes, the assurance process can work under the non-fully connected situation. Moreover, when the sensor nodes in the same group of clustered WSNs [20] cannot communicate with each other, the network must be re-grouped. It is temporary that the nodes are not fully connected. We investigate the added overhead in non-fully connected networks in Section 4.

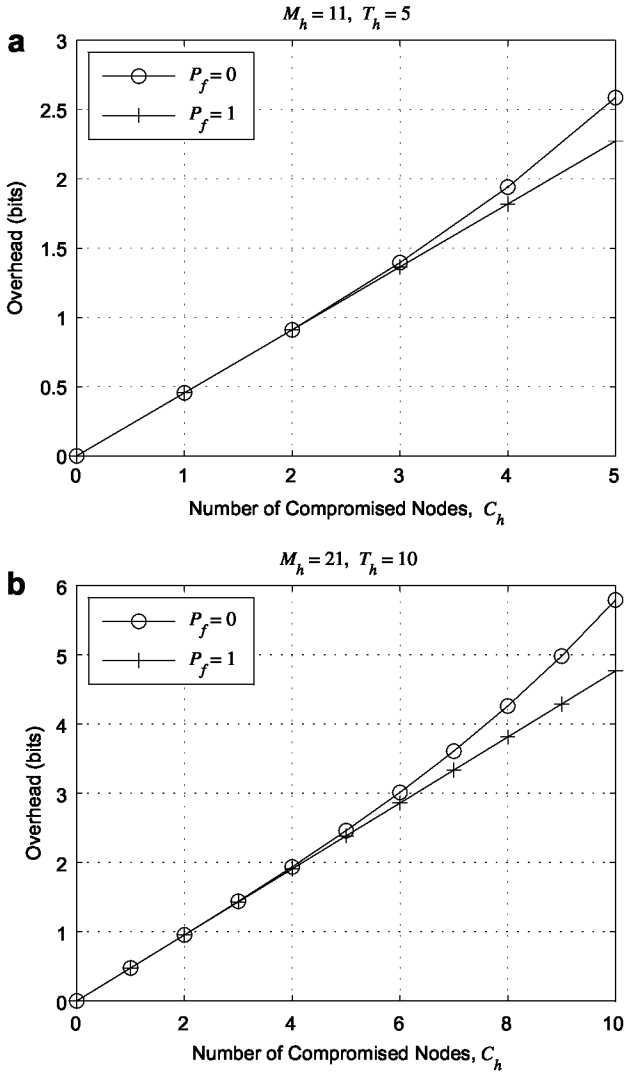

Fig. 5. The overheads in the hth hop for $P_f = 0$ and $P_f = 1$ when (a) $M_h = 11, T_h = 5$ and (b) $M_h = 21, T_h = 10$.

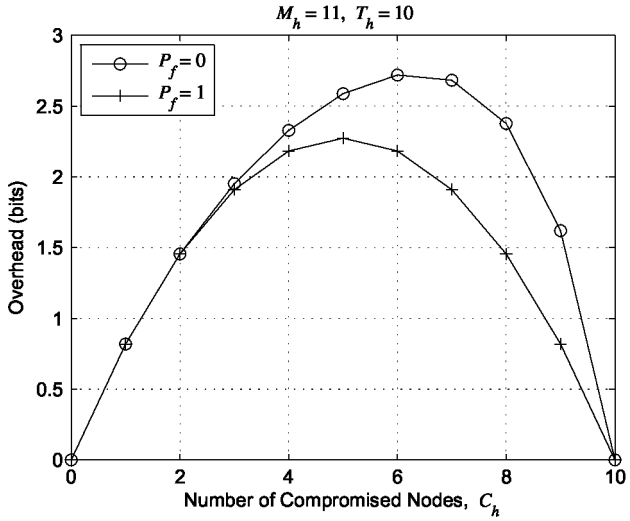

Fig. 6. The overheads in the hth hop for $P_f = 0$ and $P_f = 1$ when $M_h = 11$, $T_h = 10$.

4. Performance evaluation

Numerical calculation and simulations are conducted to evaluate the performance of the proposed algorithm. Ten thousand of Monte Carlo tests are run for each simulation. In the first set of simulations, the number of bits required for agreeing with the vth transmitted fusion result, $k_h(v)$, is the same as that given in Section 3.

Packet headers are assumed to 5 bytes (40 bits) and 10 bytes (80 bits) in simulations. When $M_h = 11$ and $T_h = 5$, the overheads of the proposed algorithm for $P_f = 0$ and 1 are illustrated in Fig. 7. Because the overhead is dominated by the header, variant P_f's have almost identical performance. The overheads are lower than 100 and 200 bits, that are $T_h/2$ times of 40 and 80 bits, respectively, when $M_h = 11$, $T_h = 5$ as illustrated in Fig. 7.

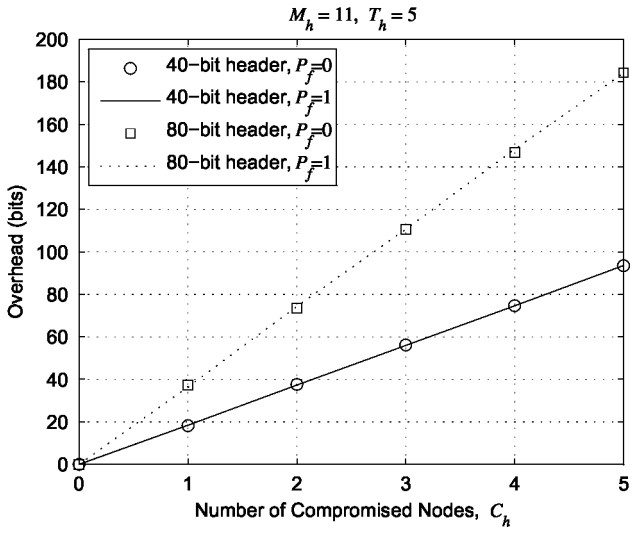

Fig. 7. The overheads of the proposed algorithm with 40-bit and 80-bit headers for $P_f = 0$ and *1* when $M_h = 11$ and $T = 5$.

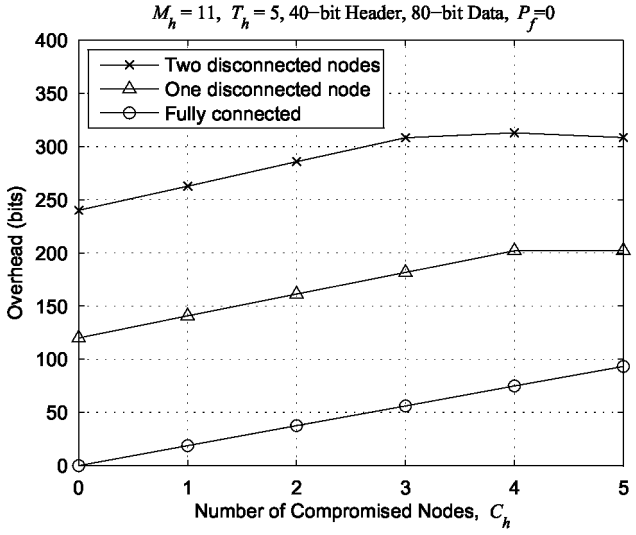

Fig. 8. The overheads of the proposed algorithm with 40-bit header and 80-bit data for a non-fully connected network and $P_f = 1$ when $M_h = 11$ and $T_h = 5$.

The nodes in the same layer of a WSN are sometimes not fully connected. That is, some fusion nodes cannot overhear the data that the other fusion nodes transmitted to the intermediate node in the upper layer. The disconnected fusion node must send its fusion result at its time slot. Fig. 8 presents the overheads when the number of disconnected nodes is zero, one, and two. Notice that the curve of the disconnected case level off because the overhead mainly comes from the disconnected nodes. Every disconnected node must send its fusion results according the algorithm. Therefore, the overhead doubles when the number of the disconnected nodes increases from one to two. However, the overhead is not lasting as explained in Section 3.4.

5. Conclusions

Wireless sensor networks are expected to be deployed to different fields to collect useful data for the base station. Due to the large number of sensors and the amount of incoming data, data fusion is often performed to reduce the overall traffic from sensors toward the base station. In this work, we investigate the information assurance issue of the data fusion process, in which compromised sensor nodes may launch stealthy attacks to trick data fusion nodes and eventually the base station to accept false results.

In this work, we have proposed a time-slotted voting scheme to achieve data fusion assurance in multi-hop fusion fashion, where data fusions are performed along the path where the results are sent toward the base station. In each of these fusion hops, a threshold T_h is set. If a fusion result is supported by more than T_h votes in the hth hop, it is accepted at the next layer. The scheme provides good security against sensor node compromise. Moreover, the traffic to be transmitted at hth hop is $O(T_h)$.

References:

[1] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, IEEE Communications Magazine 38 (8) (2002) 102–114.

[2] D. Culler, D. Estrin, M. Srivastava, Overview of sensor networks, IEEE Computer 37 (8) (2004) 41–49.

[3] [3] J.-F. Chamberland, V.V. Veeravalli, Asymptotic results for decentralized detection in power constrained wireless sensor networks, IEEE Journal on Selected Areas in Communications 22 (6) (2004) 1007–1015.

[4] D'Costa, V. Ramachandran, A.M. Sayeed, Distributed classification of gaussian space–time sources in wireless sensor networks, IEEE Journal on Selected Areas in Communications 22 (6) (2004) 1026–1036.

[5] D.L. Hall, S.A.H. McMullen, Mathematical Techniques in Multisensor Data Fusion, second ed., Artech House, Norwood, MA, 2004.

[6] Y. Lin, B. Chen, P.K. Varshney, Decision fusion rules in multi-hop wireless sensor networks, IEEE Transactions on Aerospace and Electronic Systems 41 (2)(2005) 475–488.

[7] S. Yiu, R. Schober, Nonorthogonal transmission and noncoherent fusion of censored decisions, IEEE Transactions on Vehicular Technology 58 (1)(2009) 263–273.
[8] Boulis, S. Ganeriwal, M.B. Srivastava, Aggregation in sensor networks: an energy-accuracy trade-off, Ad Hoc Networks 1 (2–3) (2003) 317–331.
[9] Z. Ye, A.A. Abouzeid, J. Ai, Optimal stochastic policies for distributed data aggregation in wireless sensor networks, IEEE/ACM Transactions on Networking 17 (5) (2009) 1494–1507.
[10] P. Sholander, A. Harris, J. Brown, Intersensor information assurance for DoD tactical networks, in: Proc. of MILCOM 2002, vol. 2, Anaheim, CA, Oct. 2002, pp. 1456–1461.
[11] Perrig, J.A. Stankovic, D. Wagner, Security in wireless sensor networks, Communication of the ACM 47 (6) (2004) 53–57.
[12] H. Chan, A. Perrig, B. Przydatek, D. Song, SIA: secure information aggregation in sensor networks, Journal of Computer Security 15 (1) (2007) 69–102.
[13] H. Chan, A. Perrig, D. Song, Secure hierachical in-network aggregation in sensor networks, in: Proc. of ACM CCS '06, Alexandria, VA, Nov. 2006, pp. 278–287.
[14] W. Du, J. Deng, Y.S. Han, P.K. Varshney, A witness-based approach for data fusion assurance in wireless sensor networks, in: Proc. of GLOBECOM 2003, vol. 3, San Francisco, CA, Dec. 2003, pp. 1435–1439.
[15] H.-T. Pai, Y.S. Han, Power-efficient data fusion assurance using direct voting mechanism in wireless sensor networks, IEEE Transactions on Computers 57 (2)(2008) 261–273.
[16] J. Deng, R. Han, S. Mishra, A performance evaluation of intrusion-tolerant routing in wireless sensor networks, in: Proc. of IPSN 2003, Palo Alto, CA, 2003, pp. 349–364.
[17] T.-Y. Wang, Y.S. Han, P.K. Varshney, A combined decision fusion and channel coding scheme for fault-tolerant classification in wireless wensor networks, IEEE Transactions on Wireless Communications 5 (7) (2006) 1695–1705.
[18] K.B. Frikken J.A. Dougherty IV, An efficient integrity-preserving scheme for hierachical sensor aggregation, in: Proc. of ACM WiSec '08, Alexandria, VA, 2008, pp. 68–76.
[19] K. Sun, P. Ning, C. Wang, Fault-tolerant cluster-wise clock synchronization for wireless sensor networks, IEEE Transactions on Dependable and Secure Computing 2 (3) (2005) 177–189.
[20] O. Younis, M. Krunz, S. Ramasubramanian, Node clustering in wireless sensor networks: recent developments and deployment challenges, IEEE Network 20 (3)(2006)20–25.
[21] K. Römer, F. Mattern, The design space of wireless sensor networks, IEEE Transactions on Wireless Communications 11 (6) (2004) 54–61.
[22] G. Mergen, Q. Zhao, L. Tong, Sensor networks with mobile access: energy and capacity considerations, IEEE Transactions Communications 54 (11) (2006) 2033–2044.

Notes:

1. We use "fusion nodes" instead of "intermediate nodes" since all sensors in the WSN might fuse data from different sources.
2. Since communication overheads for the local time synchronization could be O(1) [19] and the local synchronization is also necessary in many other applications, we will ignore the extra power consumption for the local time synchronization.
3. We are considering a general situation that the bits required for agreeing with vth transmitted fusion result might different for vs. If they are the same, then Oh(1) is simply khTh, where kh is the number of such bits.
4. A common length for an MAC is about 128 bits.