Incident-Avoidance Routing in Wireless Sensor Networks

Mohammad R. Faghani, Uyen T. Nguyen, *Member, IEEE*

***Abstract*— Wireless sensor networks (WSNs) are being deployed widely thanks to recent advances in wireless communication technologies. Many WSNs may form in hostile environments, especially in military applications. Sensor nodes are thus prone to different types of attacks such as jamming, collision attacks, and eavesdropping. Once a sensor node is compromised, it is likely that the information passing through this node will be revealed to the attacker, or will never reach the destination (e.g., in jamming attacks). In this paper, we propose a cross-layer scheme that uses information from the application layer to locate compromised nodes, computes a new, secure path connecting the source and destination and routes data packets along the new path to the destination. We present our simulation results to show the effectiveness of the proposed incident-avoidance routing algorithms.**

***Index Terms*— cross-layer design, fault tolerance, jamming attack, secure routing, wireless sensor network**

I. Introduction

WIRELESS sensor networks (WSNs) are used ubiquitously in different types of applications from surveillance and monitoring to personal health care and navigation systems. In most of WSN applications, security is a major concern. Deployed in a hostile environment, especially in military applications, wireless sensors are prone to different types of attacks such as eavesdropping, jamming and collision attacks. Once a sensor node is compromised, it is likely that the information passing through this node will be revealed to the attacker, or will never reach the destination (e.g., in jamming and black hole attacks).

Although there exist solutions to the above problem, they are not practical to be applied to WSNs. For example, one solution to jamming attacks is channel surfing [10]. Nodes under attack will switch to another channel that is not being jammed. This requires some degree of coordination among nodes to select and tune in a common channel. This type of coordination consumes energy and requires high computational power, which is not suitable for sensor nodes. Another solution attempts to overcome the jamming condition by increasing power level or using more complex coding schemes in addition to prioritization of messages that a node sends [11].This solution is also energy-consuming and computationally intensive.

In this paper, we propose a cross-layer scheme that uses information from the application layer to locate compromised nodes, computes a new, secure path connecting the source and destination, and route data packets along the new path to the destination. Sensors can take advantage of existing techniques [19, 20] to determine whether they are under attack. After that, they apply our proposed scheme for forwarding data to the destination via a secure route instead of the original path. We present our simulation and result visualization to show the effectiveness of the proposed incident-avoidance routing algorithms.

The remainder of this paper is organized as follows. We discuss related work in section II. In section III, we describe the proposed incident-avoidance routing algorithm. Simulation results are presented in Section IV. We summarize the paper in Section V.

II. Related Works

Many secure routing protocols have been proposed for ad-hoc networks [7, 8, 10-14]. However, most of them cannot be applied directly to wireless sensor networks due to computational power and energy limitations of sensor nodes.

There exist also secure routing algorithms designed for ad-hoc networks that are based on public key cryptography [10-14]. Nevertheless, public key cryptography is computationally intensive and not suitable for use in WSNs.

Routing algorithms based on symmetric key cryptography such as [15–18] are less intensive computationally. However, they are based on source routing or distance vector routing, and not appropriate for WSNs.

Greedy Perimeter State Routing (GPSR) [9] uses geographical locations of sensor nodes to establish routing paths. To route around an area through which the path cannot pass, the protocol tries to find the perimeter of the planar area. To do so, GPSR greedily sends out a packet for potentially many hops, before the packet loops and is recognized as undeliverable. The routing delay is thus potentially high in GPSR.

Our proposed algorithm, on the other hand, computes a secure path first and then sends packets along the new path, eliminating the delay caused by packet looping. Moreover, we propose a fleeting algorithm that connects the source to a backup sink when the primary sink is located inside the area under attack.

III. The Proposed Incident-Avoidance Routing Algorithm

In this section, we describe our proposed routing algorithm

which, upon being notified of the location of an area under attack, will route data away from that area. In section III.A, we provide an overview of the proposed algorithm, including the computation of virtual coordinates and selection of next-hop candidate routers. We then present the next-hop node selection policy section III.B, and a performance analysis of this algorithm in section III.C. In section III.D, we describe the fleet algorithm to be used when the primary sink is located inside the incident area.

A. Overview of the Proposed Routing Algorithm

We assume that the sensor nodes are capable of detecting anomalous activities around them. For example, wireless multimedia sensors have the video capability to detect and identify attackers [1]. Traditional sensor nodes, on the other hand, may use jamming and intrusion detection schemes [19, 20] to monitor their environment. Our proposed scheme is cross-layered in the sense that information from the application layer (video or jamming/intrusion detection confirmation) will trigger the execution of the proposed routing algorithm at the network layer.

Upon detecting an incident, a sensor sends an *event notification message* to its neighbors. This sensor is called a *tainted sensor* since there is a high probability that it is currently (or will soon be) under attack. A sensor counts the number of distinct event notification messages it received from its neighbors. The count allows the sensor to estimate how far it is from the incident. Generally speaking, the closer it is to the event, the more messages it receives.

We also assume that a receiver (sink) is associated with one or more backup receivers (sinks). When the main sink fails or is under attack, data will be routed to a backup sink. This type of deployment redundancy is a common practice in networking, for load balancing and combating node failures and security threats. Every node in the network is informed of the locations of all the sinks, and can compute its own location using a localization algorithm [9]. The node first computes the Euclidean distance to every sink, and then selects the closest node as its *primary sink* and the second closest node as the *backup sink*.

Our proposed routing algorithm uses three types of coordinates as in [4]: absolute coordinates, virtual coordinates and mapping coordinates. They are defined as follows.

Let o denote the origin of the coordinate system (e.g., the south west corner of the physical deployment area), t denote the sink, and h denotethe node on the routing path that wishes to select the next node to add to the routing path, as illustrated in Fig. 1. To initiate the next-node selection process, h broadcasts its absolute coordinate(x_h^o, y_h^o) to its neighbors. A neighbor node i thus knows the position of its upstream node h, which is (x_h^o, y_h^o), in addition to its own position (x_i^o, y_i^o) and the sink position (x_t^o, y_t^o).

The *virtual coordinates* of a node (e.g., node*i* in Fig. 1) are defined as its coordinates in the virtual two-dimensional coordinate system where its upstream node (e.g., node*h* in Fig. 1) is the origin, and the *X*-axis is the line connecting the upstream node h and the sink. In the example shown in Fig. 1, the virtual coordinates of i are denoted by(x_i, y_i), and calculated as follows:

$$\begin{cases} x_i = \cos(\theta).(x_i^o - x_h^o) + \sin(\theta).(y_i^o - y_h^o) \\ y_i = \cos(\theta).(y_i^o - y_h^o) - \sin(\theta).(x_i^o - x_h^o) \end{cases} \quad (1)$$
$$\theta = \tan^{-1}\left(\frac{y_i^o - y_h^o}{x_i^o - x_h^o}\right)$$

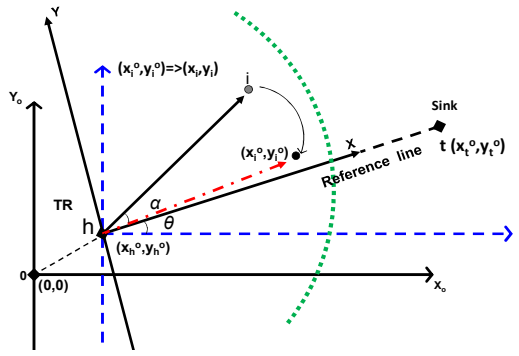

Fig. 1. Next hop selection in DGR [4]

We define *ReferenceLine* as a straight line connecting the origin of the virtual coordinate system (e.g., node*h* in Fig. 1) to the sink; *DeviationAngle* (α) is the angle that specifies how much a path is expected to deviate from the *ReferenceLine* at the origin point. If we rotate the virtual coordinates around the origin by an angleα, the rotated coordinates are called*mapping coordinates*. If $\alpha > 0$, the rotation is clockwise; if $\alpha < 0$, the rotation is counterclockwise. Moreover, $\alpha = 0$ means that the path will be the shortest path along the direction from h to the sink. The mapping coordinates are used during the path establishment process to determine a node's suitability of becoming the next node on the routing path. In the example shown in Fig. 1, the mapping coordinates of i are denoted by (x_i^m, y_i^m), and calculated as follows.

$$\begin{cases} x_i^m = \cos(\alpha).x_i + \sin(\alpha).y_i \\ y_i^m = \cos(\alpha).y_i - \sin(\alpha).x_i \end{cases} \quad (2)$$

B. Next Hop Selection Policy

To discover a *direction-aware* path, the source first broadcasts a probe message for route discovery. Theselected next node will continue to broadcast the probe message to find its next forwarding node, and so forth. A probe message contains the following information (see also Fig. 2):

- source ID (*SourceID)*
- IDs of the primary sink (*SinkID1)*and backup sink (*SinkID2)*
- sequence number of this packet (*SeqNum)*
- deviation angle α as defined above (*DeviationAngle)*. If α is anegative value, a path will be established below the *ReferenceLine* connecting the current node and the sink; otherwise, the path will be above the *ReferenceLine*.
- source-to-sink hop count H_s (field *SrcToSinkHopCount*) which is the ideal hop count from the source to the sink. Let R be the maximum transmission range of a sensor node and D_{src}^t be the

distance between the source and the sink. Therefore, $H_s = \left\lceil \frac{D_{Src}^t}{R} \right\rceil$.

The above values in a probe message are set by the source and are not changed while the probe message is traversing the network. A probe message also contains the following fields whose values will be changed by intermediate nodes on the routing path:

- the hop count from the source to the current node (*HopCount*)
- the ID of the current node which is generating this probe message (*PreviousHop*)
- the absolute coordinates of the current node (*Absolute-Position*)

Upon receiving a probe message, a node will calculate its virtual coordinates based on its upstream neighbor's position which is given in *AbsolutePosition* field of the probe message it just received. Then, mapping coordinates is calculated based on the virtual coordinates and the *DeviationAngle* using (2).

Fixed Values			
SourceID	SEQNum	SinkID1	SinkID2
Deviation Angle	SourceToSinkHopCount		
Variable Values			
HopCount	PreviousHop	Absolute Position	

Fig. 2 Information in a probe message

In Fig. 3, the point (*R*,*0*) is named the *StrategicMapping-Location*. This point is located on the *ReferenceLine* at distance R from a node *h* currently looking for a next forwarding node.In practice, it is unlikely that the next hop neighbor of *h* is located exactly at the *StrategicMappingLocation*. Hence, we select a neighbor of *h* whose mapping coordinates are the closest to the *StrategicMappingLocation* [4].

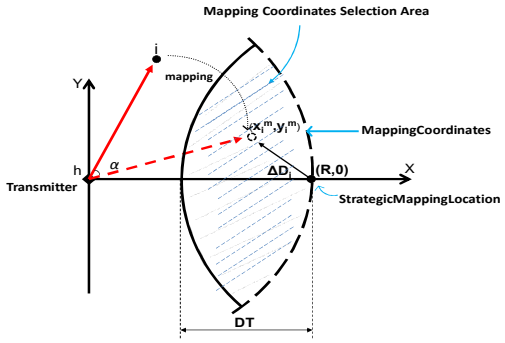

Fig. 3 Computingmapping coordinates [4]

The shaded area in Fig. 3 illustrates the neighbor selection area. The neighboring nodes whose mapping coordinates are located in this area are considered next hop candidates (*NHCs*).

Let ΔD_i be thedistance between the *StrategicMappingLocation*and the mapping coordinates of node *i* in the neighbor selection area. Thus, ΔD_i can be calculated as follows:

$$\Delta D_i = \sqrt{\left(x_i^m - R\right)^2 + \left(y_i^m\right)^2} \quad (3)$$

To limit the selection area, a threshold, namely *DT* is set. Node *i*becomes an *NHC* if $\Delta D_i < DT$. In our proposed scheme, we exclude the NHCs that are located in the incident area. Conforming to this condition, the backoff time t_b node *i* has to spend before replying to the probe message from *h* is calculated as follows [4]:

$$t_b = \tau \times \Delta D_i + rand(0, \mu), \quad (4)$$

whereτ is a fixed interval, $rand(0, \mu)$ gives a random value uniformly distributed in $(0, \mu)$ and μ is a small constant.Among the timers of the next hop candidates, the one with the smallest $t_b$value will expire first and that node becomes the next forwarding node. Note that the smaller the ΔD_ivalue, the shorter the backoff time, allowing a node closer to the *StrategicMappingLocation* to be selected. If node *h* has more than one neighbor with the same ΔD_ivalue, the random number $rand(0, \mu)$helps break the tie.

To enable the routing path to diverge from the area under attack, we proposed a new mechanism which adds an extra delay to $t_b$to extend the response time, as follows:

$$t_b = \tau \times \Delta D_i + rand(0, \mu) + n \times \gamma, \quad (5)$$

where*n* is the number of received event notification messages and γ is a constant value. The larger the value *n*, the closer the node is to the incident. Its timer is thus set longer so that it is less likely to be selected as the next forwarding node.

Let *i*be the node with the shortest timer t_b. As soon as its timer expires,node *i* will then send a unicast reply message (REP) to its upstream node *h*. To avoid collision among different REP messages at node *h*, τ is set to an adequately large value. Node *h* will only accept the first REP message and ignores the subsequent replies. Upon receiving the first REP message, *h* will broadcast a selection message (SEL) containing the ID of node *i*. All other *NHCs* hearing the SEL or REP message will stop their backoff timers. When node *i* receives the SEL message containing its ID, it generates and broadcasts a probe message, and the above algorithm is repeated until the sink receives a probe message. The sink then broadcasts a confirmation message to terminate the path establishment process.

C. *Performance Analysis*

Routing away from the incident will obviously increase the length of the path. We now compute the increase in path length. In Fig. 4, the straight line *d*represents the original path without applying our proposed scheme and the dashed arc *m*is the traversed path after applying the proposed scheme.

The values of *m* and *d* are calculated as follows:

$$m = r\theta,$$
$$d = \sqrt{r^2 + r^2 - 2r^2\cos(\theta)}, \quad (6)$$

where*r* is the radius of the incident area and $0 < \theta < \frac{\pi}{2}$is the

angle corresponding to arc *m* (see Fig. 4).

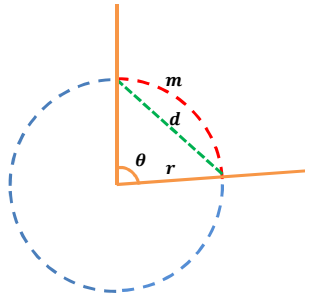

Fig. 4Computing the difference in path length

The differenceΔP_a in path length is thus:

$$\Delta P_a = r\theta - \sqrt{r^2 + r^2 - 2r^2\cos(\theta)}, \quad (7)$$

which is, on average,about 30% of the radius:

$$\frac{1}{\pi} \times \int_0^{\pi} (r\theta - \sqrt{r^2 + r^2 - 2r^2\cos(\theta)})d\theta = r\left(\frac{\pi^2}{2} - 4\right)\frac{1}{\pi} \quad (8)$$
$$\cong 0.3r$$

Although the resulting path length is longer, our proposed algorithm allows data to reach the sink securely. In many cases, it is the only resort that allows data to reach the intended destination.

D. *Fleet Algorithm*

In many cases, the primary sink node may be located inside the hazardous area where the incident is taking place. We thus propose a new scheme called "fleeting" in which data packets will be routed to a backup sink to avoid the area under attack.

To enable the capability of fleeting from the incident area, a source will select the closest sink as the primary and the second closest sink as a backup. It then includes both sinks in a probe message (fields *SinkID1* and *SinkID2*). We assume that all nodes know their geographical locations using a localization technique [9].

As a probe message reaches the boundary of the area under attack (i.e., reaches a tainted sensor or a node in the mapping coordinates selection area (see Fig. 3)), the node *N* receiving the probe message will check whether the primary sink is within the hazardous area. This can be done using one of the existing event boundary detection algorithms such as [21]. If the primary sink is inside the event boundary, node*N* sends a "go away" message to its upstream node *M* (which sent the probe message to *N*).

To avoid collisions of multiple "go away" messages at node *M*, node *N* sets a timer before sending the "go away" message. The timer t_b is computed as follows:

$$t_b = rand(0, \tau) \quad (9)$$

where$rand(0,\tau)$ gives a random value uniformly distributed in $(0,\tau)$with τ being a small constant.

As soon as node *M* receives a "go away" message, it considers itself a new source and the original source's backup sink *SinkID2*as its primary sink. It will also choose its own backup sink (e.g., *SinkID3*) and start the route discovery procedure described in section III.B to connect itself to the sink *SinkID2*. The final route will be the path from the original source *S* to *M* concatenated with the path from *M* to the backup sink *SinkID2*.

An alternative implementation to the above is to let node *M*send a "Redirect" message to the source*S*. Upon receiving it, *S* will then start a new route discovery from itself to the backup sink *SinkID2*. The path from S to *SinkID2* is more likely to be shorter than the concatenated path *S-M-SinkID2*. However, this implementation will increase the route establishment latency. If the source has only a few packets to send, the longer path *S-M-SinkID2* will allow the source to send data as soon as possible. Furthermore, if sink *SinkID2* is also inside a hazardous area (a fact node *M* will discover during its route discovery), *M* will still be able to connect *S* to another backup sink, *SinkID3*, instead of making *S* search for a second backup sink and restart the route establishment process. This, again, helps reduce the route establishment latency.

The concatenated path *S-M-SinkID2* is potentially longer than the original path from *S* to the primary sink *SinkID1*. In Appendix A, we compute the additional distance incurred by the detour route. This is the cost of routing data safely to a backup sink when the primary sink is compromised.

IV. Simulation Results

We used MATLAB for our simulations. The results are then visualized (Fig. 6 to 8) to show the effectiveness of the proposed algorithm. We conducted three sets of experiments to

a) evaluate the effectiveness of the incident-avoidance routing algorithm discussed in section III.B;
b) evaluate the effectiveness of the fleeting algorithm described in section III.D;
c) validate the performance analysis presented in section III.C.

A. *Effectiveness of the Incident-Avoidance Routing Algorithm*

In this experiment, 900 sensors are randomly distributed in an area of size 100m x100m as shown in Fig. 5. The radio range of sensor nodes is R=10m. An incident took place at coordinates (60, 44), and all sensors within a distance of 25m of the incident sensed the event. These tainted sensors are shown in the figure by red or dark colors.

A route request is received from the application layer to establish a connection from asource at coordinates (0.6, 43.7) to a destination at coordinates (98.3, 43.9), both in the purple color in Fig. 5. If the route is established with an initial deviation angle of $\frac{\pi}{2}$, the connection will go through the area under attack as shown in Fig. 6.

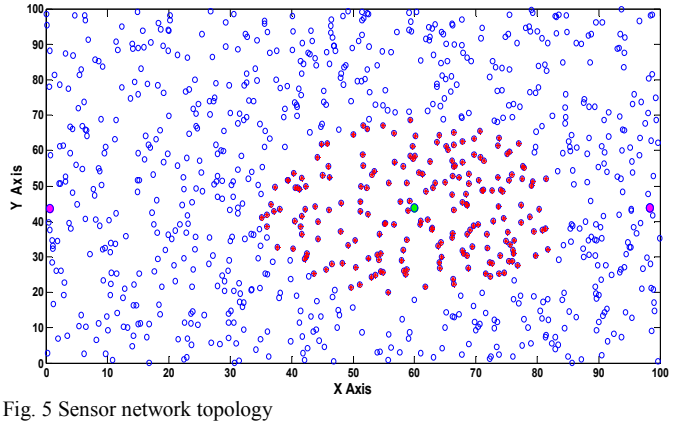

Fig. 5 Sensor network topology

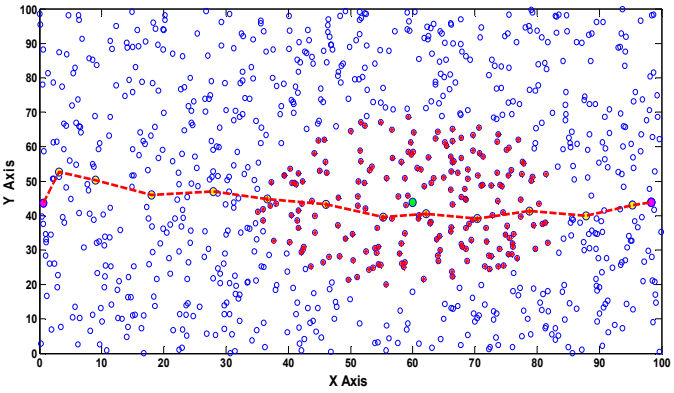

Fig. 6 Passing through incident area

Using the proposed scheme, each sensor in the area under attack will reply after a longer time-out interval. This means that sensors that are not within that area are more likely to be selected as the next hop on the routing path. Fig. 7 showsthe new routing path after the proposed routing algorithm is applied. As it can be seen, the new route bypasses the incident area to deliver data securely to the destination.

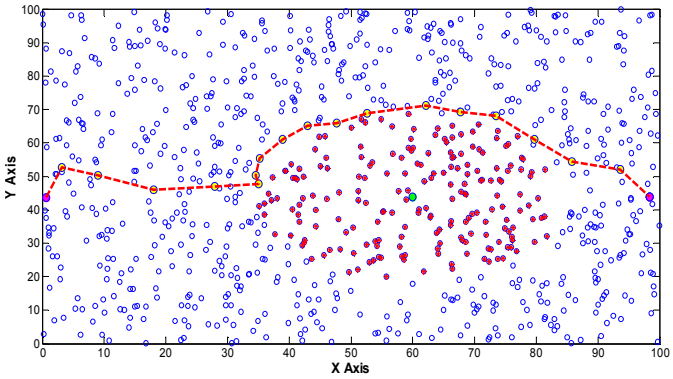

Fig. 7 Bypassing the incident area using the incident-avoidance algorithm

B. *Effectiveness of the Fleeting Algorithm*

In this experiment, we re-used the above configuration and network setting, except that the receiver (the primary sink, depicted by the asterisk in magenta color) is now inside the area under attack.

The fleeting capability of the proposed algorithm is illustrated by Fig. 8. As soon as the probe message reached the border of the incident area, it fleeted away from the area to reach the backup sink.

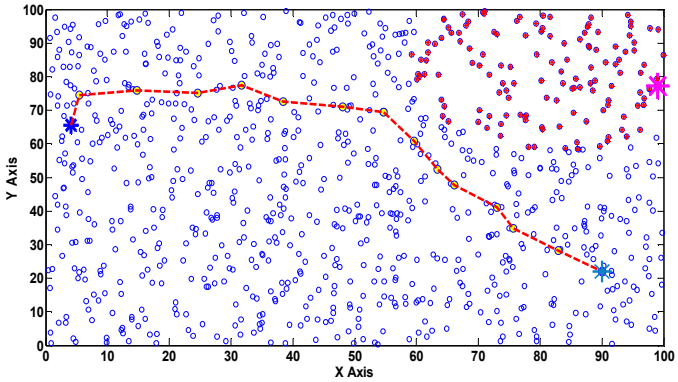

Fig. 8 Fleeting away from the incident area

Although the detour route is longer than the original, it insures that data traffic will safely reach a sink. If we assume that there is only one area under attack of size A in a network of size S, then the probability of using the fleet algorithm is A/S. If $A \ll S$, this probability is almost negligible. That is, we rarely have to use the fleet algorithm for routing.

In Appendix A, we compute the average path length of the detour route in comparison with the original route.

C. *Validation of the Performance Analysis*

We conducted the following simulation to validate the analytical result derived in section III.C. First, we selected all the sensors located on the border line of the area under attack, and divided them into two groups. One group (in cyan color) is on the left side of the area and the other (in green color), on the right side as shown in Fig. 9. The nodes in one group sent data to all nodes in the other group using our proposed incident-avoidance algorithm and the commonly used directional geographic routing (DGR) algorithm. Fig. 9 shows one example flow between two nodes: the arc is the path computed by the proposed algorithm whose length is w, and the dashed line is the path given by DGR whose length is v. We then computed the difference w-v for all possible pairs of nodes from the two groups, and took the average of all the obtained difference values.

The graph in Fig. 10 shows the results obtained from difference radii of the incident area. The x-axis represents the radius of the incident area, ranging from 10 to 40 units. The y-axis shows the average path length difference w-v normalized to the radius of the incident area. All the y values are about 0.4, which are close to the 0.3 value given by the analytical model in section III.C. The difference between the simulation results and the analytical model can be explained by the fact that the incident area in a real network is not an exact circle as assumed in the analytical model. This also explains the fact that as the incident area becomes larger, the average path length difference w-v normalized to the radius increases. The increase is about 10% as the radius expands from 10 to 40 units.

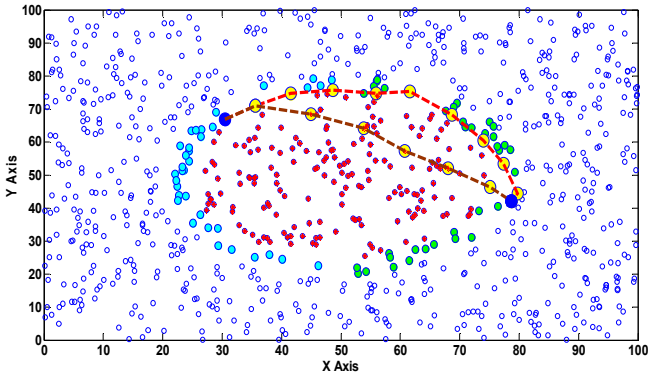

Fig. 9. Different possible path from Source to Sink

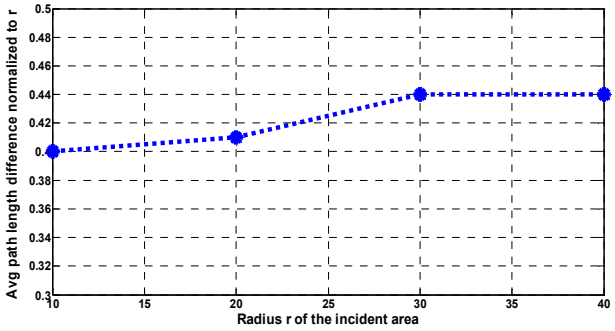

Fig.10. Average path length difference normalized over *r*

V. CONCLUSION

Many wireless sensor networks are deployed in hostile environments such as battlefields where they are vulnerable to different types of attacks. In this paper, we propose a cross-layer algorithm that reroutes data away from an area under attack. The algorithm uses information from the application layer to locate compromised sensor nodes and computes a new source-to-destination path that bypasses the compromised nodes and their surrounding area. We also propose a new algorithm called *fleeting* that addresses more serious cases in which primary sinks are located *inside* the incident area. In these cases, the algorithm uses the information of a backup sink stored in probe messages and computes a new route connecting the source to the backup sink. The route establishment latency and end-to-end delay will be more likely to increase, but this case will rarely occur if the attack is not a targeted attack as discussed earlier.Our simulations and visualization confirm the effectiveness of the proposed algorithms. Moreover, our performance analysis shows that the alternate paths are generally 30% longer than the original paths in both cases. However, that is the cost of routing data securely to the destinations when part of the network is under attack. In many cases, it is the only way to allow data to reach the destinations.

APPENDIX A

In this appendix, we compute the path length difference between the original route and the detour route given by the fleet algorithm described in section III.D.

Suppose that the detour route is divided into three segments P_1, $P2$ and P_3 as shown in Fig. 11. The circle represents the area under attack, which has radius r. The route from the source to the primary sink consists of two sub-paths: P_1 and x, which are outside and inside the incident area, respectively. P_2 is the arc to be traversed along the border of the incident area, and $P_3$is the sub-path connecting $P_2$to the backup sink. Let l be the straight line connecting the two end points of arc P_2.

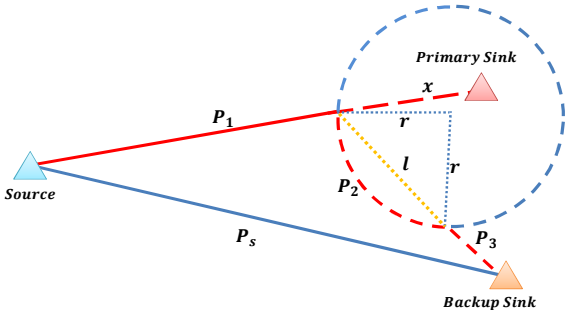

Fig. 11Routing paths in the fleet algorithm

The Euclidian distance from the source to the primary sink is P_1+x, while the length of the detour route given by the fleet algorithm is equal to P_1+P_2+P_3. We compute the difference ΔP_f between the lengths of the detour and original routes.

$$\Delta Pf = P_1 + P_2 + P_3 - P_1 - x = P_2 + P_3 - x$$

Given that

$$\begin{aligned} P_2 &= 2r\theta \\ l + P_3 &= \sqrt{{P_1}^2 + {P_s}^2 - 2P_1P_s\cos(\gamma)} \\ l &= 2rsin(\theta) \\ P_3 &= l + P_3 - l \\ P_3 &= \sqrt{{P_1}^2 + {P_s}^2 - 2P_1P_s\cos(\gamma)} - 2rsin(\theta) \end{aligned} \tag{10}$$

we obtain

$$\Delta P_f = 2r\theta + \sqrt{{P_1}^2 + {P_s}^2 - 2P_1P_s\cos(\gamma)} - 2rsin(\theta) - x$$

whereP_1, P_s, θ, γ and x are independent random variables between (0, k-x), (0,k), $(0, \frac{\pi}{2})$, $(0, \frac{\pi}{2})$ and $(0, 2r)$, respectively and$k$ is the length of the longest possible path (a straight line) in the network. For instance, if the network is a rectangular, then k is the length of the diagonal.

Note that there is no explicit formula to compute the average value of ΔP. However, as discussed in section IV.B, if we assume that $A \ll S$, there is only one area under attack, and the attacker uniformly chooses the area, then the probability of having the primary sink in the incident area is negligible. As a result, the expected value of the additional path length incurred by the incident-avoidance algorithm (section III.B) or the fleet algorithm (section III.D) is as follows:

$$\overline{\Delta P} = \left(1 - \frac{A}{S}\right) \times \overline{\Delta P_a} + \frac{A}{S} \times \overline{\Delta P_f} \approx 0.3r \quad (11)$$

where ΔP_a is given by equation (7).

REFERENCES

[1] M. Guerrero-Zapata, R. Zilan, J. M. Barcelo-Ordinas, K. Bicakci, and B. Tavli, "The Future of Security in Wireless Multimedia Sensor Networks: a position paper,"Springer Telecomm. Syst. Journal, Vol.45, No.1, pp.77, 2010.

[2] M. R. Faghani, and S. M. A Motahari, "Sectorized Location Dependent Key Management," *In Proc. of WIMOB 2009*. pp.388-393, 12-14 Oct. 2009

[3] Kundur, D.; Luh, W.; Okorafor, U.N.; Zourntos, T., "Security and Privacy for Distributed Multimedia Sensor Networks," In *Proc. of the IEEE* , vol.96, no.1, pp.112-130, Jan. 2008

[4] M. Chen, V. C. M. Leung, S. Mao, and Y. Yuan, "Directionalgeographicalrouting for real-time video communications in wireless sensor networks," *Computer Communications*, vol. 30, no. 17, pp. 3368–3383, 2007.

[5] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of Service Attacks in Wireless Networks: The Case of Jammers," *Communications Surveys & Tutorials, IEEE* , vol.13, no.2, pp.245-257, Second Quarter, 2011

[6] D. Malan, M. Welsh, and M. Smith. A public-key infrastructure for key distribution in Tinyos based on elliptic curve cryptography. In *1st IEEEInternational Conference on Sensor and Ad Hoc Communications and Network*s, pages 71–80, October 2004.

[7] K.Sanzgiri, B.Dahill, B. N. Levine, Clay Shields, E. M. Belding-Royer. A secure routing protocol for ad hoc networks. In proceeding of *IEEE International Conference on Network Protocols (ICNP)*, 99(99), November 2002.

[8] M. Guerrero Zapata and N. Asokan. Securing ad hoc routing protocols. In Proc. *ACM Workshop on Wireless Security (WiSe),* pages 1–10. ACM Press, 2002.

[9] A. Srinivasan and J. Wu, "A Survey on Secure Localization in Wireless Sensor Networks," *Encyclopedia of Wireless and Mobile Comm*., CRC Press, Taylor and Francis Group, 2007.

[10] W. Xu, W. Trappe, Y. Zhang, "Channel surfing: defending wireless sensor networks from interference" In*Proceedings of the6th international conference on Information processing in sensor networks*, pp. 499–508. ACM, New York, NY, USA (2007)

[11] W. Xu, K. Ma,W. Trappe, Y. Zhang, "Jamming sensor networks: attack and defense strategies". *Network, IEEE* , vol.20, no.3, pp. 41- 47, May-June 2006

[12] L. Zhou, Z. Haas, "Securing ad hoc networks", *IEEE Network Magazine* 13 (6) (1999) 24–30.

[13] F. Stajano, R.J. Anderson, "The resurrecting duckling: security issues for ad-hoc wireless networks", In *proceeding ofSeventh International Security Protocols Workshop*, 1999, pp. 172–194.

[14] J. Hubaux, L. Buttyan, S. Capkun, "The quest for security in mobile ad hoc networks", In *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing* (MobiHOC 2001), 2001.

[15] Y.-C. Hu, D.B. Johnson, A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks", In *Proceedings of the4th IEEE Workshop on Mobile Computing Systems and Applications* (WMCSA 2002), 2002, pp. 3–13.

[16] Y.-C. Hu, A. Perrig, D.B. Johnson, "Ariadne: secure on demand routing protocol for ad hoc networks", in *proceeding ofMOBICOM*, 2002.

[17] S. Basagni, K. Herrin, E. Rosti, D. Bruschi, "Secure pebblenets", In *proceeding ofACM International Symposium on Mobile Ad Hoc Networking and Computing* (MobiHoc 2001), 2001, pp. 156–163.

[18] P. Papadimitratos, Z. Haas, "Secure routing for mobile ad hoc networks", In *proceeding ofSCS Communication Networks and Distributed Systems Modeling and Simulation Conference* (CNDS 2002), 2002.

[19] W. Xu, W. Trappe, Y. Zhang, T. Wood, "The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks," In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing* (Mobihoc), 2005

[20] G. Thamilarasu, A. Balasubramanian, S. Mishra, R. Sridhar, "A cross-layer based intrusion detection approach for wireless ad hoc networks", In *proceeding ofMobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on* , vol., no., pp.7 pp.-861, 7-7 Nov. 2005

[21] K. Ren, K. Zeng, W. Lou, Fault-tolerant event boundary detection in wireless sensor networks, In *proceeding of* IEEE GLOBECOM, vol. 7, 2006, pp. 354- 363.

Mohammad R. Faghani (M'05) received B.Sc. degree in communication engineering and M.Sc. in communication network engineering from Isfahan University of Technology, Isfahan, Iran, in 2006 and 2009 respectively. He is currently working toward the PhD degree since 2010. His research interest includes, wireless network security, online social network security and web malware.

Uyen T. Nguyenreceived her Bachelor of Computer Science and Master of Computer Science degrees in 1993 and 1997, respectively, from Concordia University, Montreal, Canada. She completed her Ph.D. degree at the University of Toronto, Canada, in 2003. From 1995 to 1997 she was a software engineer at Nortel Networks, Montreal, Canada. She joined the Department of Computer Science and Engineering at York University, Toronto, Canada, in 2002 and is currently an Associate Professor. Her research interests are in the areas of mobile computing, wireless networking, multimedia applications and network security. Prof. Nguyen has published about 30 papers in refereed international journals and conference and workshop proceedings. She is on the editorial board of Communications Journal published by ACTA Press. She is serving as a program co-chair for the 3rd International Conference on Mobile, Ubiquitous, and Intelligent Computing (MUSIC 2012). She is also a program vice-chair of the 13th IEEE International Conference on High Performance Computing and Communications (HPCC 2011) and the 7th International Conference on Future Information Technology (FutureTech 2012).