International Journal of Modern Trends in Engineering and Research
www.ijmter.com
e-ISSN No.:2349-9745, Date: 2-4 July, 2015

Review on RCDPWSN: Reliable approach for Cut Detection and Prevention in Wireless Sensor Networks

Prof. Priti Subramanium[1],Mr.Vijay J. Chaudhari[2]

[1]*Assistant Professor, Department of Computer Science and Engineering, Shri Sant Gadge Baba College of Engineering and Technology, Bhusawal,pritikanna559@gmail.com.*
[2]*M. E. Student, Department of Computer Science and Engineering, Shri Sant Gadge Baba College of Engineering and Technology, Bhusawal, viju.ch14@gmail.com*

Abstract—In a Wireless Sensor Network, sensor nodes may fail for many reasons, such as hostile counteracting,intrusion,electrical or mechanical problems etc. A wireless sensor network can get separated into multiple network connected components due to the failure of some of its component nodes that is called a "cut". An algorithm is proposed for detection that allows (i) every node to detect when the connectivity to a specially designated node has been lost, and (ii) one or more nodes (that are connected to the neighbor node after the cut) to detect the occurrence of the cut.The RCDPWSN system can very easily prevent these cuts in wireless sensor networks.
Keywords - hostile counteracting, intrusion, cut, wireless sensor network, PROBE massage.

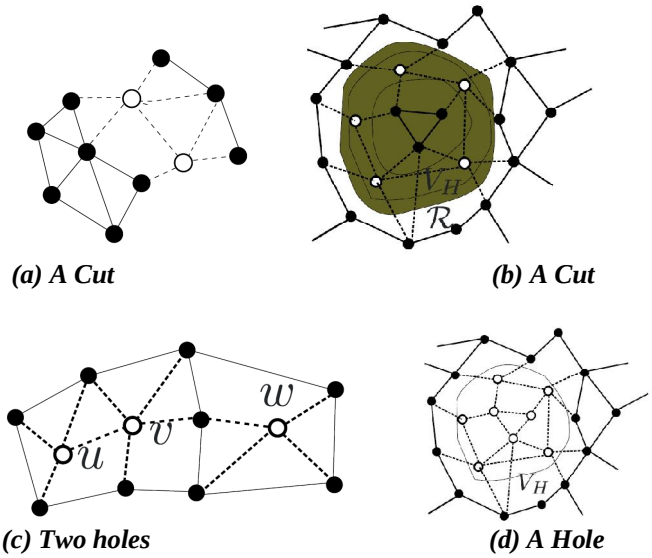
I. INTRODUCTION

A wireless sensor network (WSN) of spatially distributed autonomous sensors to monitor physical, mechanical or environmental conditions and to cooperatively pass their sensory information through the wireless network to a main location. The more modern networks are multi-hop, also able to control of sensor activity itself. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance; today such networks are used in many industrial and consumer applications, such as industrial process monitoring, ship monitoring, animal migration monitoring, medical monitoring, vehicle monitoring and, health monitoring, and so on.
We consider the problem of detecting cuts by the nodes of a wireless network. We consider that there is a specially designated node in the network, its call the source node. Since a cut may or may not separate a node from the source node, we differentiate between two outcomes of a cut for a particular node. When a node u in the network is disconnected from the source, we say that a Disconnected frOm Source (DOS) event has arises for u. When a cut occurs in the network that does not separate a node u from the source node, we say that Connected, but a Cut Occurred Somewhere (CCOS) event has occurred for u node. By detection of cut we mean

1) Detection by each node of a DOS event when it occurs, and
2) Detection by the nodes close to cut of CCOS events and the approximate location of the cut.

II. DISTRIBUTED CUT DETECTION

Our problem is divided into two parts. First, we want to enable every node to detect if it is disconnected from the source (i.e., if a DOS event has occurred). Second, we want to enable nodes that are close to the cuts but are still connected to the source (i.e., those that experience CCOS events) to detect CCOS events and alert the source node. There is an algorithm-independent limit to how accurately cuts can be detected by nodes still connected to the source, which are related to holes. Fig. 1 shows example.



(a) A Cut *(b) A Cut*

(c) Two holes *(d) A Hole*

Figure.1 Example of cuts and holes. Dark circles represent active nodes and unfilled circles represent failed nodes. Solid lines represent edges, and dashed lines represent edges that existed before the failure of the nodes. The hole in (d) is indistinguishable from the cut in (b) to nodes that lie outside the region R.

We focus on developing methods to distinguish small holes from large holes/cuts [3]. We allow the possibility that the algorithm may not be able to tell a large hole (one whose circumference is larger than 'max) from a cut, since the examples of Figs. 1b and 1c show that it may be impossible to distinguish between them[4]. Note that the discussion on hole detection part is limited to networks with nodes deployed in 2D.

III. DOS DETECTION USING DCD ALGORITHM

The approach here is to exploit the fact that if the state is close to 0 then the node is disconnected from the source, otherwise not. In order to reduce sensitivity of the algorithm to variations in network size and structure, use a normalized state. DOS detection part consists of steady-state detection, normalized state computation, and connection or separation detection. Every node i maintain a binary variable $\widehat{DOSi}$ (K), which is set to 1 if the node believes it is disconnected from the source and 0 otherwise. This variable, which is called the DOS event status, is initialized to 1 since there is no reason to believe a node is connected to the source initially. A node maintains track of the positive steady states seen in the past using the following method. Each node i computes the normalized state difference $\partial x_i(k)$ as follows:

$$\partial \mathrm{xi(k)} = \begin{cases} \dfrac{x_i(k) - x_i(k-1)}{x_i(k-1)}, & if\, x_i(k-1) > \varepsilon_{zero} \\ \infty, otherwise \end{cases}$$

Where ε_{zero} is a small positive number. A node i maintains a Boolean variable PSSR (Positive Steady State Reached) and updates PSSR(k) ← 1 if $|\partial \mathrm{xi(k)}|\varepsilon_{\Delta x}<$ for k = k $-\tau_{\mathrm{guard}}$, k $-\tau_{\mathrm{guard}}$ + 1, . . . , k (i.e., for τ_{guard}consecutive iterations), where$\varepsilon_{\Delta x}$ is a small positive number and τ_{guard} is a small integer. The starting 0 value of the state is not considered a steady state, so PSSR(k) = 0 for k = 0, 1, ... τ_{guard}. Each node keeps an estimate of the most recent "steady state" observed, which is denoted by $\hat{x}_i^{ss}(k)$. This estimate is updated at every time k according to the following rule: if

PSSR(k) = 1, then $\hat{x}_i^{ss}(k) \leftarrow x_i(k)$, other-wise $\hat{x}_i^{ss}(k) \leftarrow \hat{x}_i^{ss}(k-1)$. It is initialized as $\hat{x}_i^{ss}(0) = \infty$. Every node i also keep a list of steady states seen in the past, one value for each unpunctuated interval of time during which the state was detected to be steady. This information is kept in avector $\hat{x}_i^{ss}(k)$, which is initialized to be empty and isupdated as follows: If PSSR(k)=1 but PSSR(k-1) = 0, then$\hat{x}_i^{ss}(k)$ is appended to $\hat{x}_i^{ss}(k)$ as a new entry. If steady statereached was detected in both k and k - 1 (i.e.,PSSR(k) = PSSR (k-1)=1), then the last entry of $\hat{x}_i^{ss}(k)$ isupdated to $\hat{x}_i^{ss}(k)$. For instance, for the node *v* in thenetwork,$\hat{x}_v^{ss}(3)$ =∅(empty),$\hat{x}_v^{ss}(60)$=(0:0019) and $\hat{x}_v^{ss}(150)$=(0:019; 0:012). For futureuse, we also define an unsteady interval for a node *i*, whichis a set of two local time counters $(k_i^{(1)}, k_i^{(2)})$ such that thestate$x^i(k_i^{(1)} - 1)$is a steady-state (i.e.PSSR $(k_i^{(1)} - 1)$) but$x^i k_i^{(1)}$is not, and $x^i k_i^{(2)}$is not steady but $x^i(k_i^{(2)} = 1)$is.

Each node computes a normalized state $\hat{x}_i^{norm}(k)$ as:

$$x_i^{norm}(k) := \begin{cases} \dfrac{x_i(k)}{\hat{x}_i^{ss}(k)}, & if\ \hat{x}_i^{ss}(k) > 0 \\ \infty, & otherwise \end{cases}$$

Where$\hat{x}_i^{ss}(k)$ is the last steady state seen by *i* at *k*, i.e., thelast entry of the vector$\hat{x}_i^{ss}(k)$. If the normalized state of *i* isless than εDOS, where εDOS is a small positive number, thenthe node declares a cut has taken place:$\widehat{DOS} \leftarrow 1$. If thenormalized state is ∞, meaning no steady state was seenuntil *k*, then $\widehat{DOSi}$ (K) is set to 0 if the state is positive (i.e.,$x_i k > \varepsilon_{zero}$) and 1 otherwise.

IV. CCOS DETECTION

The algorithm for detecting CCOS events relies on finding ashort path around a hole, if it present, and is minimally inspired by the jamming detection algorithm proposed in [4]. The method utilizes node states to assign the task of hole-detection to the most appropriate nodes. When a nodedetects a maximum change in its local state as well as failure ofone or more of its neighbors, and both of these events occurwithin a (predetermined) small time interval, the a PROBE message is initiated by node. There is following information in each PROBE message p:

- A unique probe ID,
- Probe centroid Cp
- Destination node,
- Path traversed (in chronological order), and
- The angle traversed by the probe around thecentroid.

The probe is forwarded in a manner such that if the probe istriggered by the creation of a small hole or cut (withcircumference less than 'max), the probe traverses a patharound the hole in a counter-clockwise (CCW) direction andreaches the node that initiated the probe. In that case, the netangle traversed by the probe is full 360 degree [5]. On the other hand, if the probe was initiated by the occurrence of a boundary cut, even if the probe eventually goes its node of initiation, the total angle traversed by the probe is zero. Nodes forward a probe only if the distance traveled by the probe (the number of hops)is smaller than a threshold value 'max. Therefore, if a probe is initiated due to a large internal cut/hole, then it will be absorbed by a node (i.e., not forwarded because it exceeded the distance threshold constraint), and the absorbing node declares that a CCOS event has taken place [6].

V. RCDPWSN SYSTEM ARCHITECTURE

The architecture of cut detection and prevention in wireless sensor network is shown in figure 2.It consist of three components:

1. Cut detection
2. Recovery
3. Prevention

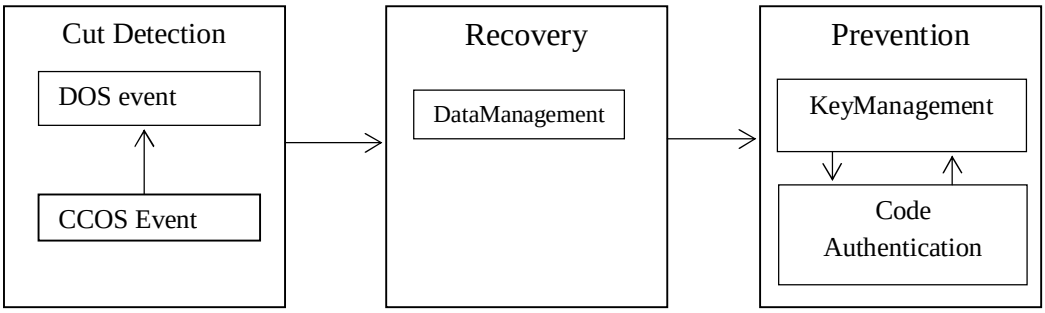


Fig 2 RCDPWSN System Architecture

5.1 Cut Detection

Imagine that a sensor that wants to send data to the source node has been disconnected from the source node. Without the knowledge of the network's disconnected state, it may simply forward the data to the next node in the routing tree, which will do the same to its proceeding node, and so on. However, this message passing merely not utilizesprecious energy of the nodes; the cut prevents the data from reaching the destination. Otherwise, if a node were able to detect the occurrence of a cut, it could wait for the network to be repaired and eventually reconnected, which saves on-board energy of numerous nodes and delaystheir lives. On the other hand, the ability of the source node to detect the occurrence and location of a cut will allow it to undertake network fixes. Thus, the capacity to detect cuts by both the disconnected nodes and the source node will lead to the increase in the operational lifetime of the network as whole. A method of repairing a disconnected network by using mobile nodes .for detecting cuts, as the one proposed here, can serve as useable tools for above network repairing methods.

A distributed algorithm to detect cuts, named the Distributed Cut Detection (DCD) algorithm. The algorithm allows each node to find DOSevents and a subset of nodes to finds CCOS events. The algorithm we propose is distributed and asynchronous: it involves only local communication between sides by side nodes, and is robust to temporary communication failure between node pairs. A key component present in

the DCDalgorithm is a distributed iterative computational step through which the nodes compute their (fictitious) electrical potentials. The convergence rate of the computing isindependent of the size and structure of the network. The DOS detection of nodes part of the algorithm is applicable toarbitrary networks; a node only needs to communicate ascalar variable to its side by sides. The CCOS detection part ofthe algorithm is limited to networks that are deployed in 2DEuclidean spaces, and nodes need to know their ownplaces. The position information need not be highlycorrect.

5.2 Recovery

The recovery of failure nodes in wireless sensor network is process of management of data, secured keys and identification of nodes. It is necessary to recover the failure nodes after the detection for to maintain the multihop paths between the nodes in the wireless sensor network.

5.3 Secured Prevention

The broadcast nature of the transmission medium in wireless sensor networks makes information more vulnerable than in wired applications. Security mechanisms such as encryption and authentication are essential to protect information transfers. Existing network security mechanisms are not feasible in this domain, given the limited processing bandwidth, storage, and power and energy resources. Public-key algorithms, such as RSA, Diffie- Hellman are undesirable, as they are computationally costly.

To develop security mechanisms and protocols for sensor networks, a necessary requirement is key management, i.e., the maintenanceand establishment of shared keys between pairs of communicating nodes. However, to bootstrap secure communications between sensor nodes, i.e., assigning secret keys among them can become a tricky task. If we were known about which nodes would be in the same neighborhood before deployment, keys could be decided apriori. Unfortunately, most sensor network deployments are random; therefore such a priori knowledge doesn't exist. There are also some other requirements that need to be considered while designing a key management protocol. A desirable feature is resistance to capturing node. Though a node is compromised and its key material is revealed, an adversary should not be able to gain control of other parts of the network by using this material. So the compromise of nodes should result in a breach of security that is constrained within a little, localized part of the network.

CONCLUSION

In this paper we proposed RCDPWSN, a Reliable approach for Cut Detection and Prevention in Wireless Sensor Networks. RCDPWSN didn't use any communication model. The DCD algorithm detects Disconnected from Source if occurred. CCOS detects and estimate the approximate location of the cut in the form of alist of active nodes that lie at the boundary of the cut/hole. Finally the prevention of cut can be done using key management and code authentication. So RCDPWSN is very sophisticated technique for detecting and preventing cuts in WSN.

REFERENCES

[1] Prabir Barooah, Harshavardhan Chenji, RaduStoleru, and Tama´sKalma´r-Nagy "Cut Detection in Wireless Sensor Networks", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 23 pp 1-8 2012.
[2] G. Dini, M. Pelagatti, and I.M. Savino, "An Algorithm for Reconnecting Wireless Sensor Network Partitions," Proc. European Conf. Wireless Sensor Networks, pp. 253-267, 2008.
[3] N. Shrivastava, S. Suri, and C.D. To´ th, "Detecting Cuts in Sensor Networks," ACM Trans. Sensor Networks, vol. 4, no. 2, pp. 1-25, 2008.
[4] H. Ritter, R. Winter, and J. Schiller, "A Partition Detection System for Mobile Ad-hoc Networks," Proc. First Ann. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (IEEE SECON '04), pp. 489-497, Oct. 2004.
[5] M. Hauspie, J. Carle, and D. Simplot, "Partition Detection in Mobile Ad-Hoc Networks," Proc. Second Mediterranean Workshop Ad-Hoc Networks, pp. 25-27, 2003.
[6] P. Barooah, "Distributed Cut Detection in Sensor Networks," Proc. 47th IEEE Conf. Decision and Control, pp. 1097-1102, Dec. 2008.
[7] S.S. Ahuja, S. Ramasubramanian, and M.M. Krunz. Single-link failure detection in all-optical networks using monitoring cycles and paths. IEEE/ACM Transactions on Networking, 17(4):1080–1093, 2009.
[8] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture Directions for Networked Sensors," Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS), 2000.
[9] Klempous R.; Nikodem J.; Radosz, L.; Raus, N. Byzantine Algorithms in Wireless Sensors Network, Wroclaw Univ. of Tech-nol., Wroclaw; Information and Automation, 2006. ICIA 2006. International Conference on, 15-17 Dec. 2006, pages :319-324

