TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 05/09 pp364-375
Volume 20, Number 4, August 2015

Topology Adaptation for Robust Ad Hoc Cyberphysical Networks under Puncture-Style Attacks

Ying Liu* and Wade Trappe

Abstract: Many cyber physical networks will involve ad hoc deployments utilizing peer-to-peer communications. Examples include transportation systems where a group of moving cars communicate in order to avoid collisions, teams of robotic agents that work together in support of disaster recovery, and sensor networks deployed for health-care monitoring, monitoring the operation of a factory plant or to coordinate and actuate mechanisms for energy conservation in a building. These networks may face a variety of threats that puncture their connectivity and, should their performance degrade, the result could be catastrophic. Consider, for example, a vehicular ad hoc network where communication assists collision avoidance. In such a case, degradation could lead to vehicle accidents. Therefore, in order to overcome network performance degradations and the puncture of a network (such as blackhole or jamming) which is under attack, we propose an algorithm called the Fiedler Value Power Adjustment Topology Adaption (FVPATA). FVPATA aims to dynamically adapt an ad hoc network's topology, even if the attacker varies its location and in the case of an interference-style attack by increasing the interference power. The algorithm utilizes the formulation from the graph theory which works with the Fiedler value to guide each node in wireless ad hoc network utilizing power adjustments to enhance the network's overall robustness. The advantage of the proposed mechanism is that it is a light-weight approach which is totally distributed, based on topology updates inherent in the Optimized Link State Routing (OLSR) protocol and, hence, it is unnecessary to introduce additional messages. Additionally, an algorithm was developed to resolve problems involving asymmetric links that arise in ad hoc networks by eliminating unnecessary energy consumption of Fiedler nodes. Simulation results using NS3 show that the proposed mechanism successfully decreases the average amount of hops used by 50% and the delay of flows when nodes are migrating at a modest rate below 60 m/min.

Key words: algebraic connectivity; Fiedler value; topology adaptation; distributed power control; ad hoc networks

1 Introduction

Many cyberphysical systems will be deployed using ad hoc wireless technologies, involving autonomous entities such as robots maneuvering in an environment. Such wireless networks can be easily subjected to a variety of attacks primarily because the transmission medium is an open one, allowing for observation and introduction of interference or false messages. These problems are particularly pernicious in the case of ad hoc networks where nodes in the network communicate with each other in a dynamic and opportunistic manner. In ad hoc cyberphysical networks, a malicious attacker can simply employ interferences to cause legitimate nodes around him unable to communicate with the neighboring nodes. Alternatively, he could

• Ying Liu and Wade Trappe are with the Department of Electrical and Computer Engineering, Rutgers University, New Brunswick, NJ 08901, USA. E-mail: yingliu@winlab.rutgers.edu; trappe@winlab.rutgers.edu.

$*$ To whom correspondence should be addressed.

Manuscript received: 2015-05-01; revised: 2015-06-18; accepted: 2015-06-28

also introduce attacks that would puncture the network by dropping packets or locally disrupting the routing procedure. In either case, a region of the network becomes unusable and the performance of the network significantly degrades around the areas near the attack.

There are many other complementary tools that can be used to cope with such attacks directed against ad hoc networks. These countermeasures are grouped into four main categories: (1) carefully design routing protocol to re-route packets around the attack area[1–4] and those attack areas can be discovered by machine learning methods[5–7]; (2) implement multi-path plus tunneling to add redundancy to the current route[8–12]; (3) adjust the location of network nodes[13–15]; and (4) apply robust and redundant coding[16–20].

In this paper, we examine a complementary approach to coping with jamming attacks in a distributed fashion in an ad hoc network. Our approach aims to aid the participants in an ad hoc network to avoid holes punctured in the network connectivity by an attacker through network control algorithms, and to strengthen the reliability of communication should the attacker shut down one or more of the legitimate nodes. In order to accomplish this, we propose an algorithm that aims to control the network topology so it can minimize network degradation in the instances of an attack. Our proposed Fiedler Value Power Adjustment Topology Adaption (FVPATA) algorithm is integrated with the popular OLSR routing protocol for wireless ad hoc networks and it uses the concept of "Algebraic Connectivity" of a network's topology, as characterized by the Fiedler value[21–23], to identify connectivity-sensitive nodes in the ad hoc network. These nodes then adjust their transmission power to enhance the network's robustness. The advantages of FVPATA are: (1) It only requires an adjustment to the power employed by a small set of carefully chosen Fiedler nodes. Thus, this method conserves energy for the whole network rather than increasing the power for all nodes. Additionally, increasing the power of every node can introduce undesirable interference, which often results in a decrease of the network throughput. (2) It is a distributed algorithm involving only local actions to affect the entire operation of the network. Each node uses a unique network topology shared by hello and TC messages of the OLSR protocol, and it tailors the topology through its own local actions.

This paper is organized as follows: In Section 2, we provide the background and mathematical foundation for our algorithm. In Section 3, we describe the mechanism of our Fiedler value power adjustment algorithm in details and demonstrate how the algorithm can be integrated with the OLSR protocol through pseudo codes. Indeed, our approach can universally be integrated with any state-sharing ad hoc network routing algorithms[24–27]. In Section 4, we analyze the performance of our FVPATA through simulations involving scenarios of different attacks, followed by our conclusion in Section 5.

2 Background and Theoretical Foundation

2.1 Attack model

As a starting point for our discussion, we shall consider a very simple attack model where an attacker is positioned near the center of a network since nodes in the central area are surrounded by densely populated neighboring nodes and could potentially become a bottleneck in traffic flows. As an example, we illustrate a network with an attack in Fig. 1. In this figure, a single attacker is located near an area in which many routes intersect, and the attacker can potentially cause serious structural damage to the network's topology by attacking one or more nodes nearby. Our approach works well with multiple attackers as our approach is distributed and the power adjustment is done according to local views of topology. However, for the sake of our discussion, we consider the case of only one MAC-layer

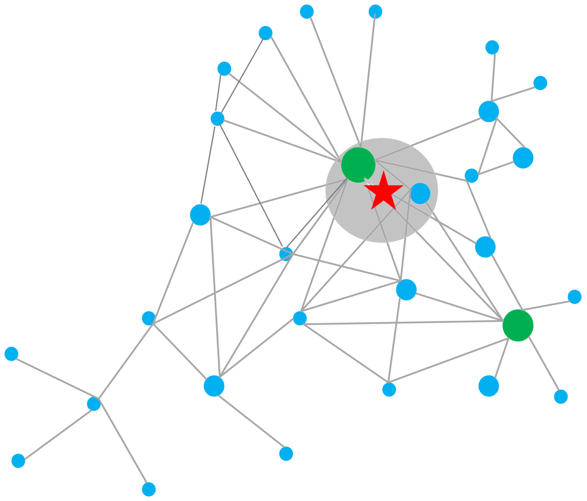


Fig. 1 A possible scenario of attacks in an ad hoc network. The red star represents the attacker placed in an area where many routes and flows must transit through. The shaded region indicates a region that is "punctured" by the attacker to effectively isolate many nodes which causes serious structural damage to the network's topology.

attacker causing harm to several neighboring nodes simultaneously.

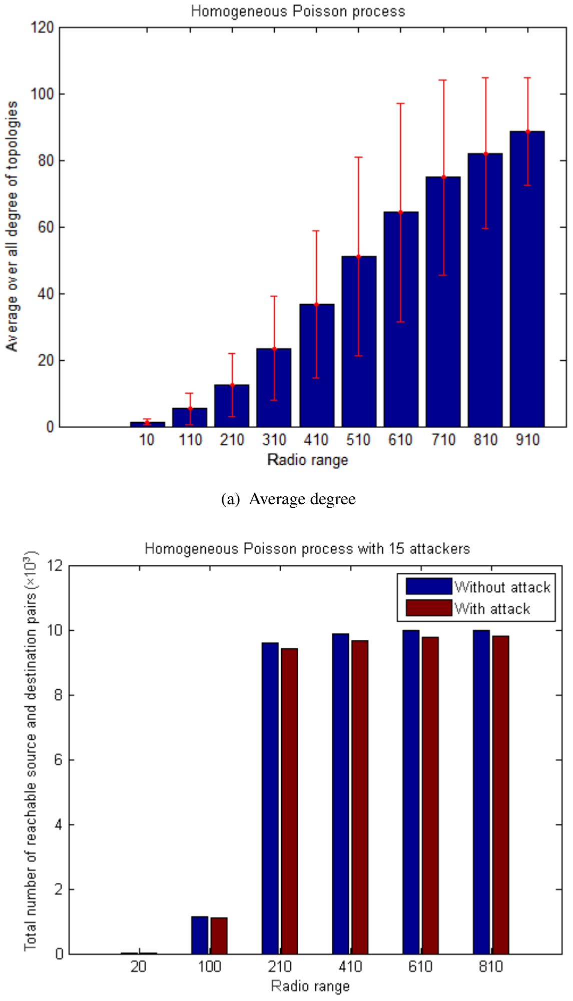
An attacker's goal is to shut down the maximum number of nodes with a minimum amount of effort. Therefore, the attacker's objective is to place himself in an area with heavy network traffic. In our example, the MAC-layer attacker continuously injects format-compliant packets to legitimate nodes without time gaps in between the packets. As a result, nodes under attack become unable to communicate properly (e.g., access the channel and complete packet transmission and reception successfully) and essentially become shut-out from the network's operation. Throughout our discussion, we will refer to an ad hoc network that uses the OLSR protocol[28]. The reason why OLSR is used is because it can support our algorithm easily since: (1) its TC message can deliver link connectivity status from three hops away and it can assist nodes in gaining complete knowledge of the network connectivity; (2) it reduces the need for extra messages when updating topology information; and (3) it is amenable for executing a distributed algorithm on each node in any ad hoc network. We note that our approach can apply equally well to other routing algorithms which have similar state-sharing features[29]. We can integrate our algorithm with them in two ways: (1) Utilize the periodic hello messages to carry the extra topology information which is from three or more hops away, such as the hello messages in Dynamic Source Routing (DSR)[24] and Ad Hoc On-Demand Distance Vector (AODV)[25]. (2) Adopt the self-contained topology update mechanism in routing protocols, for example, Global State Routing (GSR)[26] consults the vectors of link states exchanged with routing information to obtain the global knowledge of the network topology, and the On-Demand Packet Forwarding Scheme (ODPFS)[27] constructs a virtual backbone among nodes. During the construction, the global topology information is propagated.

2.2 Motivating foundation

The main purpose of FVPATA is to increase the network's robustness while minimizing the energy needed to confront an attack. The network's robustness/connectivity is closely related to the node's degree in a network graph[30]. The average node degree in a random network (when being deployed according to a spatial homogeneous Poisson process) increases with the node's radio range, as illustrated in Fig. 2a. Moreover, Fig. 2a also indicates that the spread or variation of the average degree first rises and then decreases at a larger radio range. Figure 2b implies that increasing the radio range can cause the number of mutually reachable source and destination pairs to grow, which directly corresponds to the graph's connectivity. However, after reaching a certain threshold no obvious growth occurs. This characteristic illustrates that it is not necessary to increase radio range indefinitely in order to strengthen the network connectivity. On the other hand, increasing transmission power without restraint can inadvertently generate unnecessary radio interference in the ad hoc network.

2.3 Fiedler value and graph-theoretic connectivity

Now we briefly introduce the graph theory and lemmas that will be applied in our algorithm, followed



(a) Average degree

(b) Connectivity

Fig. 2 The relationship among radio range, node degree, and connectivity.

by a description on the working mechanism behind FVPATA.

Network connectivity can be depicted by its algebraic connectivity, also called the Fiedler value. It is the second smallest eigenvalue of the Laplacian matrix, $\boldsymbol{L}(V, E)$, of a network's topological graph, $G(V, E)$, where V is a vertex set and E is the edge set connecting two vertices in graph. The Fiedler value is always non-negative, and its amplitude is proportional to the graph connectivity. It is zero if and only if the graph is disconnected. The number of zero eigenvalues in the eigenvalue set of $\boldsymbol{L}$ equals to the number of connected components in a graph. According to Ref. [21], the Fiedler value represented by λ_1, of a graph, G, can be obtained by the following eigenvalue optimization problem.

$$\lambda_1 = \min \boldsymbol{y}^{\mathrm{T}} \boldsymbol{L}(V, E) \boldsymbol{y},$$
$$\text{s.t.}\ \boldsymbol{y}^{\mathrm{T}} \boldsymbol{y} = 1 \wedge \boldsymbol{y}^{\mathrm{T}} \mathbf{1} = 0 \tag{1}$$

where $\boldsymbol{y}$ is a vector which does not equal to $\mathbf{1}$.

The Laplacian matrix of a given graph is defined as follows: Given a graph $G(V, E)$ without self cycles and multiple links between two nodes, the Laplacian matrix $\boldsymbol{L}$ is calculated by

$$\boldsymbol{L}(V, E) = \boldsymbol{D}(V, E) - \boldsymbol{A}(V, E) \tag{2}$$

where $\boldsymbol{D}(V, E)$ is a diagonal matrix whose diagonal entry contains the degrees for each node. $\boldsymbol{A}(V, E)$ is the adjacency matrix with each entry being a value of zero or one when nodes are connected to each other. In addition, its diagonal is zero since $G(V, E)$ has no self cycles. According to Eq. (2), the Laplacian matrix has the following properties[31, 32]:

- **Lemma 1**: $\boldsymbol{L}$ is a symmetric matrix. Its (i, j)-th and (j, i)-th entries are same and its diagonal entries contain each node's total degree.
- **Lemma 2**: All its eigenvalues are real since $\boldsymbol{L}$ is symmetric.
- **Lemma 3**: $\boldsymbol{L}$ is a positive semi-definite matrix. Thus, it has no negative eigenvalues. Its first smallest eigenvalue is always 0 since the sum of each row or column is zeros. By sorting the eigenvalues, we obtain: $\lambda_0 = 0 \leqslant \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_{n-1}$.
- **Lemma 4**: The number of zeros in eigenvalue indicates the number of disconnected components in the graph. If the graph is strongly connected, then the second smallest eigenvalue λ_1, which is also the Fiedler value, is always larger than zero.
- **Lemma 5**: If the attacker kills the links in between nodes or when the network's links are broken because of natural distances, the Fiedler value $\lambda_1(V, E_1) \leqslant \lambda_1(V, E)$, where $E_1 \subseteq E$.
- **Lemma 6**: Fiedler value's upper bound is limited to the minimum degree of nodes and the total number of nodes exists in networks. The upper bound approaches to the minimum value of degrees in nodes when the network is large. The exact relationship between them is given by Ref. [31].

$$\lambda_1(V, E) \leqslant \frac{|V|}{|V| - 1} \min_v d_v \tag{3}$$

Lemma 5 informs us that the Fiedler value can become larger when adding edges to a graph. Thus, a network becomes more robust as the Fiedler value increases, which implies stronger connectivity.

Our objective is to identify the weakest point in the network connection and heuristically improve the network by increasing the degree or number of neighbors associated with that node. Particularly, we are interested in what happens when we remove a node from a network's graph, and hence we will introduce a modified notion of the Fiedler value, which corresponds to the impact associated with removing all of a node's links (i.e., connections to other nodes in the network). Specifically, we define a node's Fiedler value as:

Definition: For a graph $G(V, E)$, the node Fiedler value associated with node j corresponds to the Fiedler value $\lambda_1(V, E_1)$, where E_1 corresponds to a revised set of edges for G where all edges containing node j have been removed from E.

With this definition in mind, we can re-examine the connectivity of the topologies that remain on a case-by-case basis after removing each node, and discover the nodes in the network whose deletion would have the most harmful impact on the network's algebraic connectivity. We propose a heuristic for improving the network's condition whereby we attach more links to the nodes with low nodal Fiedler value.

Lemma 6 informs us that increasing the degrees for all nodes is ineffective because the upper bound of Fiedler value is constrained by the minimum value of the degree of the nodes. Conceptually, we only need to select a few nodes to add links to, and this will be reflected in the FVPATA algorithm by having each node examine whether it is in the set of m nodes with the lowest Fiedler value.

3 OLSR-Based Topology Adaptation Algorithms

In this section, we use the Fiedler value's properties previously described to guide an online cross-layer power adjustment scheme that enhances the network's robustness when facing an attack. The idea is to select a node that is least-suitably connected to the network. By increasing the power of transmission on those nodes, we can strengthen network's capability by being able to reach more nodes and thus improve the network's overall connectivity.

3.1 Choose the node

As the amplitude of the Fiedler value represents the network's connectivity, we choose several nodes in accordance with their Fiedler values after determining their associated adjacency matrices with those nodes removed. Removing a node from the adjacency matrix means deletion of the corresponding i-th row and column.

Each node first builds an adjacency matrix for the topology it obtained in an online manner from OLSR's TC and hello messages, which are sent periodically by OLSR protocol. In the following section, the procedure for obtaining topology from the OLSR protocol will be discussed. This topology is updated every T time-units where T is a free parameter that can be adjusted. Upon obtaining the adjacency matrix, a node calculates a list of Fiedler values from the remaining adjacency matrices through removing each node. A node with the smallest Fiedler value indicates that deletion of that particular node will cause maximum damage to the network.

Moreover, nodes with the least number of links often correspond to being located in a less densely populated area, or in an area without many surrounding neighbors. They could also be located in an environment where the condition of the local channel is poor with large levels of local noises making them likely to be Fiedler nodes. In these situations, increasing the nodes' power might be inefficient due to a significant amount of energy being needed to reach other nodes or to overcome the channel conditions. However, to give them the opportunity to connect to a larger network, our algorithm is iterative in the sense that it continuously chooses the weakest nodes from the resulting network topology. Each node in the minimum Fiedler value set will choose to increase its power with probability p. We choose p according to the binomial distribution, in terms of n_1, where n_1 is the number of nodes with the least number of neighbors and N is the network size. Hence, p equals to $1-\left(1-\dfrac{1}{N}\right)^{k(N-n_1)}$ and k is a parameter that manages the tradeoff between adding power and redundancy while k can vary for each node.

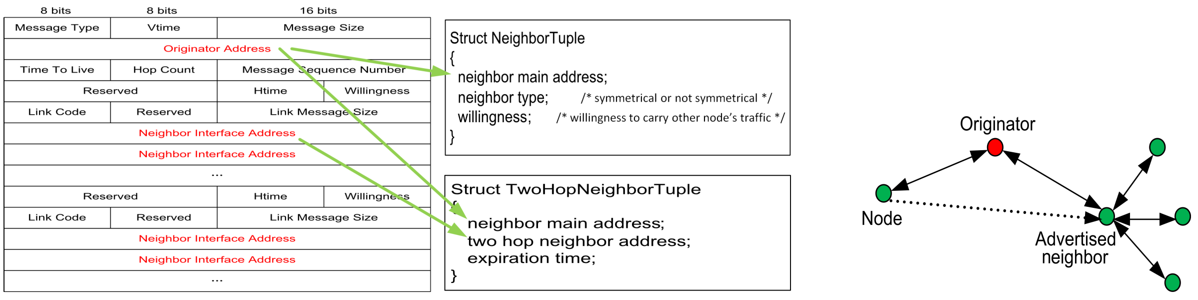
Upon obtaining the self-evaluated Fiedler node id, a node ascertains whether this node id is identical to its own. If so, it starts to increase transmission power until reaching the degree or power limit. Otherwise, it recalculates the Fiedler value using the remaining adjacency matrices. The remaining adjacency matrices are obtained by deleting the row and the column of each corresponding Fiedler node id that was obtained from the previous round. This process iterates until reaching a maximum number of iterations, or a node becoming a Fiedler node, whichever happens first.

3.2 Getting the updated topology

The adjacency matrix for the network's topology is critical when calculating the Fiedler value. We obtain it from the TC and hello messages of the OLSR protocol.

3.2.1 Hello messages and obtaining one- and two-hop neighbors

Hello messages in OLSR protocol provide both one and two hops of neighboring link status information. Messages not received by directly connected neighbors are discarded. Figure 3 shows the hello message format and contains the link status information from the network topology. The link code in the hello message identifies both the link and the neighbor type between the originator and its following list of neighbor interfaces. When receiving hello messages, the originator's main address is stored into the neighbor's main address in the neighbor tuple as shown in Fig. 3a. The originator is the node's one-hop neighbor if the main address of "Neighbor Interface Address" field is the address of the node itself. The rest of the main address of the "Neighbor Interface Address", whose neighbor type is symmetric specified by the link code, corresponds to the node's two-hop neighbors that are intermediately connected by the originator. This two-hop neighbor's main address is stored in the two-hop neighbor's tuple, shown in Fig. 3a. Figure 3b illustrates the mapping from the protocol field to the network topology.



```
Struct NeighborTuple
{
  neighbor main address;
  neighbor type;        /* symmetrical or not symmetrical */
  willingness;      /* willingness to carry other node's traffic */
}
```

```
Struct TwoHopNeighborTuple
{
  neighbor main address;
  two hop neighbor address;
  expiration time;
}
```

(a) Hello message

(b) Two-hop neighbors

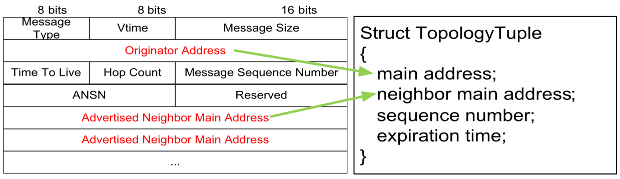
Fig. 3 Hello message and data structure storing one or two-hop neighbors.

3.2.2 TC messages for obtaining more than two-hop neighbor link status

Since a node only receives one-hop neighbors' hello messages, we are unable to obtain link statuses on nodes that are more than two hops away from hello messages. Therefore, we had to adopt TC messages to solve this issue. When receiving TC messages, a node first verifies whether the sender of TC messages is from the set of trusted one-hop neighbors. If not, it discards the message. If yes, it updates the TC tuple shown in Fig. 4 if and only if the Advertised Neighbor's Sequence Number (ANSN) is larger than the previous one stored. Else, it adds a new TC tuple if there is no record found. The "originator" field in the OLSR message is then copied to the main address and the Advertised Neighbor's Main Address is also copied to the neighbor's main address shown in Fig. 4.

3.2.3 Integrating Fiedler value into the OLSR protocol

Therefore, the OLSR protocol gives the opportunity for a node to construct self-evaluated adjacency matrices to support the ability for each node to calculate Fiedler values that represent the network connectivity. Whenever a hello or TC message is received, the OLSR protocol processes the messages and stores the link connections in each corresponding tuples. At each time interval, a node re-computes the adjacency matrices according to their neighboring table. If nodes have symmetric links between them, then the corresponding entry of the adjacency matrice is set to be one, otherwise, it is zero. The diagonals of the adjacency matrice are set to be as zeros since a node has no self cycle. However, a node is able to set the exploration time for itself. One exploration corresponds to removing a connectivity-weakest node from the adjacency matrix. This exploration process iterates until the connectivity-weakest node becomes the node itself or when the total exploration time is reached. As soon as a node realizes it is the connectivity-weakest node, it begins to increase its transmission power if its degree is below the total degree limit and its transmission power is lower than the total power limit. The node stops increasing its transmission power within a certain time. This process is online and distributed and the pseudo code is given in Algorithm 1.



```
Struct TopologyTuple
{
  main address;
  neighbor main address;
  sequence number;
  expiration time;
}
```

Fig. 4 TC message and data structure of storing neighbors.

Algorithm 1: Increase the transmission power of Fiedler nodes until each of them reaches the degree limit

```
Initialization degree_limit = $D$ and power_limit = $P$
Every time slot:
Get adjacency_matrix from the OLSR protocol by hello and TC messages
Get $n_1$ from adjacency_matrix and set explore to be zero
while $\left[1-\left(1-\frac{1}{N}\right)^{\text{explore}\cdot(N-n_1)}\right] <$ threshold do
    for  i from 0 to num_nodes−1  do
        remove node i from adjacency_matrix
        calculate the second smallest eigenvalue
        fiedler_list.push_back(the second smallest eigenvalue)
    end for
    /∗ get the connectivity-weakest node ∗/
    Sort(fielder_list)
    /∗ index 0 refers to the smallest fielder value ∗/
    node_to_adjust = fielder_list(0);
    if node_to_adjust = self id then
        if self degree< $D$ and self power < $P$ then
            increase power of node_to_adjust
            break
        end if
        adjacency_matrix.remove(node_to_adjust)
        explore=explore+1
    end if
end while
```

3.3 Symmetric and asymmetric links

In FVPATA, we only consider symmetric links to meet the requirements of the OLSR routing table so that we can guarantee packets are successfully delivered. However, asymmetric links may occur. For example, one Fiedler node may raise its transmission power in order to get connect with another node. However, that node might remain at the same transmission power if that node's total degree limit has already been reached or that node is not a Fiedler node. In this case, the Fiedler node continues to increase its transmission power and wastes the energy oblivious to the other node having no willingness/incentive to cooperate. For instance, in Fig. 3b, the advertised neighbor already has four neighbors. If the degree limit is three, it has no incentive to increase its power to connect with the node. Thus, an asymmetric link may exist between a Fiedler node and a non-Fiedler node, as shown in the dotted line in Fig. 3b. Further, the Fiedler node will iteratively increase its power until its total power limit is reached.

To solve the problem caused by asymmetric links, we require each non-Fiedler node to verify whether the Fiedler node has enough neighbors before increasing its power. After n time intervals, a non-Fiedler node verifies whether the Fiedler node is an asymmetric neighbor. If yes, then it examines whether the Fiedler node is a two-hop neighbor of itself and whether the number of Fiedler node's neighbors remains under its degree limit. If the answer to both conditions is yes, this non-Fiedler node will increase its power if the power limit for itself has not been reached. This process iterates until the Fiedler node becomes its symmetric neighbors or when the number of Fiedler node's neighbors have reached the degree limit, whichever comes first. The link code in hello message can indicate whether the link is asymmetric or symmetric with the symmetric links being its neighbors. Algorithm 2 gives the pseudo code when dealing with asymmetric issues.

Algorithm 2: Solve asymmetric links of Fiedler node

```
After $n$ time slots:
if node is not Fiedler node but the obtained Fiedler node is its asymmetric neighbor then
    if Fiedler node is node's two hop neighbors then
        calculate the number of Fiedler node's neighbors from self-evaluated adjacency_matrix
        if Fiedler node degree $< D$ and self power $< P$ then
            increase non-Fiedler node power
        end if
    end if
end if
```

Our power adjustment algorithm has weaknesses and does not solve all the problems. Firstly, since the beacons in the OLSR protocol may update the topology too fast, it may cause an overshoot of some nodes' degrees and may result in larger interference due to too much transmission power. Secondly, the algorithm does not totally solve the asymmetric link problem although we designed the algorithm to allow non-Fiedler nodes to respond to the connection request of Fiedler nodes. However, for a node which is in a low density area, even if some nodes reply to its request, the final degree still cannot meet the degree requirement and that node continues to increase transmission power until exhausting all the energy.

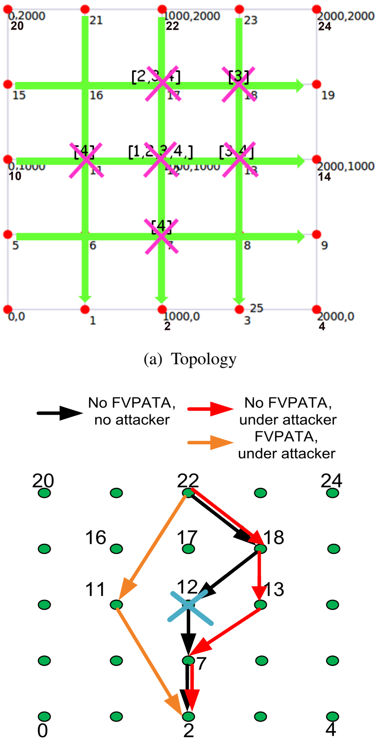
4 Simulation

4.1 Simulation results

Simulations were performed using the NS3 network simulator. The topology used in the simulations is shown in Fig. 5a. There are a total of 25 nodes positioned in a grid with an interval of 500 meters in between nodes both horizontally and vertically. Exploring the case of a grid topology gives us a clear view on the operation of our schemes. In the study, we introduced a total of six flows running through the networks simultaneously, shown as green lines in the Fig. 5a. As we can see, those flows are close to each other and this creates interferences among themselves.

Beyond the existence of interferences among flows, we introduced an attacker who can simultaneously shut down several nodes near its location. We also assumed the attacker's power would grow gradually so that he can affect one node to five nodes. We examined the effectiveness of FVPATA in terms of average hop, delays, and throughputs under different attacking scenarios. The results show that FVPATA has provided a significant improvement in performance when the nodes are under these different kinds of attacks.

We placed the attacker at four different locations, corresponding to four scenarios where they are all approximately centered around the populated area as shown in Fig. 5a. The purpose behind this attack was to simulate the attacker's attempts to reach out to as many flows as possible. In the first scenario, we assumed the attacker had a relatively low power level based on its



(a) Topology

(b) True path

Fig. 5 Topology, flow pattern, attacker position, and true paths. (a) Numbers in parentheses refer to different jamming locations. "1" denotes a MAC-layer attacker at Node 12. "2" denotes a MAC-layer attacker between Nodes 12 and 17. "3" depicts a MAC-layer attacker located at the intersection of diagonal lines of Nodes 12, 13, 17, and 18. "4" refers to a MAC-layer attacker located at Node 12 with a radio range of 500 meters. (b) Changes of true paths with one node failure with or without FVPATA.

radius of impact; hence, he could only attack one node in the network. Therefore, we chose Node 12, which is the center of the whole topology. For the second attacking scenario, the MAC-layer attacker raised its transmission power with an effective attacking range of 250 meters. We placed the attacker in between Node 17 and Node 12 to block the source transmission as much as possible. For the third scenario, we increased the attacker's radio range to be $\sqrt{2} \times 250$ meters and position it at the intersection of two diagonal lines of a rectangle composed by Nodes 12, 17, 13, and 18. In the last scenario, we set the attacker's radio range to 500 meters and put it in the center of the flows again. Under this condition, the attacker could affect Nodes 12, 11, 17, 13, and 7. Under all four scenarios, the attacker's interference range increases so as the attacker's ability to affect other nodes.

The simulation runs contained 1000 time-units and our application data started at 100 time-units after the routing tables were established and continuously fed into the network without gaps in time. The attacker started the attack at 100 time units while the OLSR protocol was running continuously to maintain the routing information. Hello messages for the OLSR protocol were sent every 2 time units and TC messages were sent at every 5 time-units when RTS/CTS was turned off. FVPATA started when it detects a 80.5775% packet loss ratio and explored at most five times in each time interval.

Figure 5b shows the changes of the route from source Node 22 to destination Node 2 after suffering from an attack with the application of FVPATA. Here we consider Scenario 1 (where only Node 12 is the node under attack) as the simplest case for an easier understanding on the concept. In Fig. 5b, the black route is the general case under the absence of an attacker and FVPATA. The route walks through Node 12, travels through Node 18, and finally reaches Node 2 which is used as a baseline for comparison. The red line is a route depicting the aftermath of shutting down Node 12 by an attacker without application of FVPATA. As the figure suggests, the red route skips Node 12 and the OLSR protocol finds an alternative route running through Node 13. Although the OLSR protocol has self-recovery capabilities, the disabled Node 12 causes traffic congestion around the affected nodes which resulted in more network-layer interference. This condition becomes even worse when more than one node breaks down. Therefore, we cannot solely rely on the self-recovery mechanism of the OLSR protocol. The brown route represents the condition when FVPATA is applied after nodes detected an abnormal packet loss. The figure shows that Node 2 increased its transmission power since it is the Fiedler node. Moreover, to obtain a degree of 6, a non Fiedler node (Node 11) also increases its transmission power and as a result, Node 2 could directly reach Node 11. Thus, the final route contained only two hops and it reduced the switching time spent by Nodes 6 and 16. FVPATA has effectively diminished the total transmission time by cutting down the number of hops needed for the flows. Figure 6 also demonstrates that FVPATA actually lowers the delay of the whole path.

While examining Fig. 6, it is clear that it illustrates the decrease in the average number of hops and the mean delays among six total flows after the application of FVPATA while the network is under attack. The

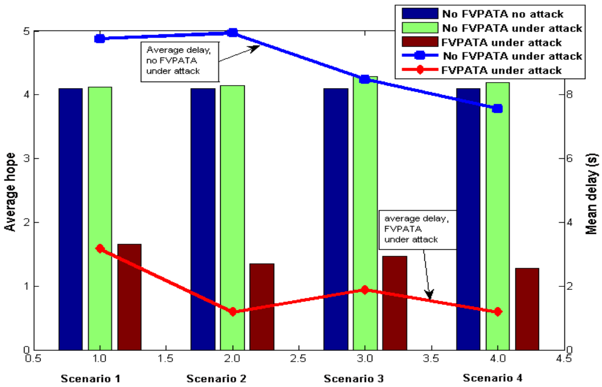


Fig. 6 Average hops and mean delay of all flows.

improvement is apparent and despite more and more nodes being shut down (Nodes 7, 11, 12, 13, and 17) in Scenario 4, its average number of hops and mean delay are the lowest. The possible explanation for this is that nodes around the attack areas begin to have reduced connections to the networks. Therefore, with large probability, removing them can cause network to be disconnected. For example, in our simulation topology shown in Fig. 5a, removing Nodes 2 and 5 can cause a separation of networks into two parts. FVPATA chooses those nodes near the border and increases their transmission power and weaves a connection between them. Therefore, if more and more nodes are disconnected, nodes near them (may also be two hops away) are more likely to be selected by FVPATA. Based on this and FVPATA's distributed structure, FVPATA should work well in situations where multiple attackers exit. In the simulations, we actually saw fewer numbers of hops when more flow interference exists.

Figure 7 represents the throughput we collected at each time interval for Scenario 1. We applied a sliding window with a width of 60 time-units to smooth data across time intervals since the network data collected experienced random variations. Simulation results indicated FVPATA increased the network throughputs compared to non-FVPATA employed network. Moreover, FVPATA converged after approximately 700 time-units and stabilized thereafter. However, the convergence time depends on the number of Fiedler nodes participating in the power adaptation. A much more disconnected network requires a longer stabilization time.

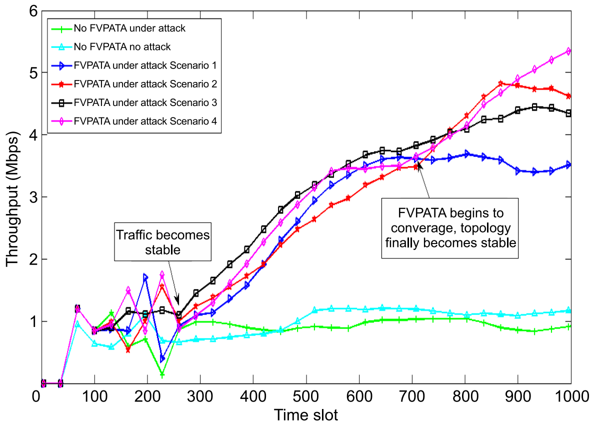


Fig. 7 Throughput for each time slot of Scenario 1.

We also computed the improvement on performance associated with the application of FVPATA under attack in Fig. 8. Comparing to the cases without power adjustments, FVPATA reduces the number average hops by at least 50% and cuts the mean delay by at least 60%. To calculate mean throughputs, we collected data from 700 and 1000 time-units during the periods where the FVPATA algorithm converges and the network throughputs stabilize. The calculated mean of the associated sample data is then compared with the case without power adjustments. We discovered there is an improvement by more than 2.5 times in terms of throughputs.

4.2 Simulation results involving mobility

Besides static networks, we also considered FVPATA's performance when facing the mobility of nodes, as might occur in a cyberphysical application involving robotic agents. Each node traveled within a square area of 2000 meters by 2000 meters, randomly changing direction every two seconds. We set each node as having the same rate and we increased their rates at succeeding rounds of simulation. In the simulations, we assumed a MAC-layer attacker tracked a node as a target and never changed its target throughout the course of attacks. The

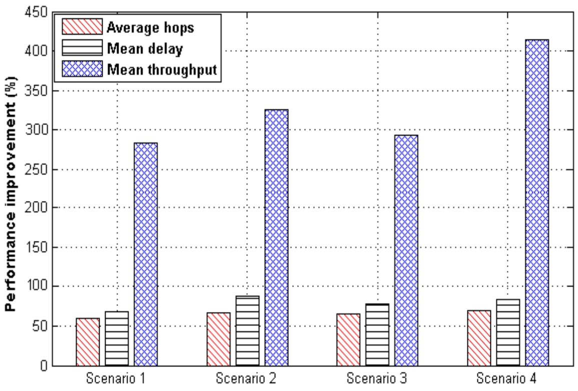
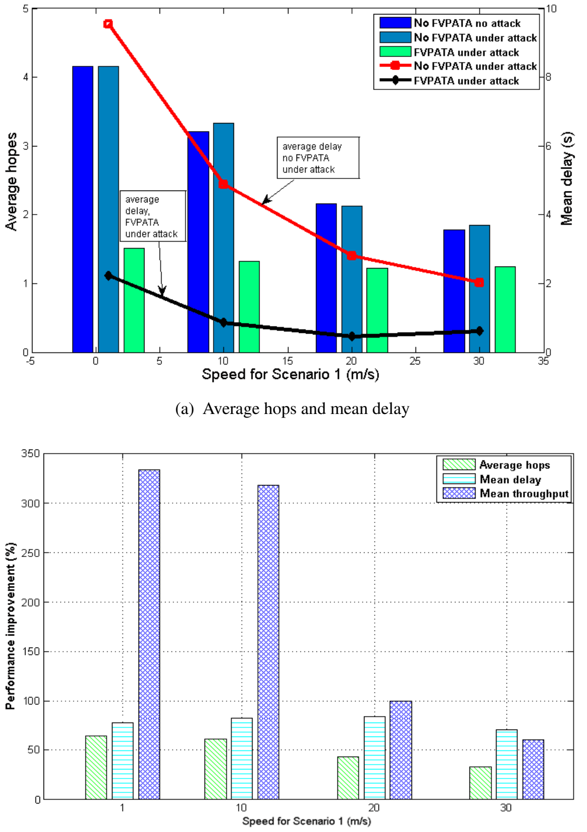


Fig. 8 Performance improvement in terms of average hop, delay, and throughput.

MAC-layer attacker could only track one node at a time since legitimate nodes moved constantly and randomly, from one place to another. Therefore, it resembles the effects in Scenario 1 we studied earlier, but in a dynamic sense. We collected the average numbers of hops, mean delays, and throughputs data under moving rates of 1 m/s, 10 m/s, 20 m/s, and 30 m/s. Each simulation ran for 1000 time-units, which is the same as static network.

Figure 9a shows that FVPATA decreases the average number of hops and mean delays when nodes are moving at moderate speeds. It also implies that sometimes increasing speed can help latency. A possible explanation for this is that some pairs of source and destination nodes could approach each other, closer and closer, to shorten the distance and time during the packet transmission. Figure 9b presents the percentage improvement on the performance of FVPATA in average number of hops, mean delays, and throughputs. We can see that the improvements of mean delay and throughputs are above 50% when the moving speed is no more than 30 m/s. The percentage improvement on average hops and throughputs decreases when the speed of nodes increase, especially the throughputs, it drops rapidly under a relative high speed. To put the speed of 30 m/s into perspective, it is equivalent to a car traveling at speed of 67 mph on the road. Therefore, the conceptual application of FVPATA in daily life may be viewed as rather practical.



(a) Average hops and mean delay

(b) Performance improvement

Fig. 9 FVPATA's performance under mobility. (a) FVPATA decreases the average number of hops and mean delays under moving nodes; (b) Even with moving node, FVPATA improves their average number of hops, mean delays, and throughputs.

4.3 Parameters that affect performance

The magnitude of FVPATA's performance improvement depends on many factors: (1) The beacons in OLSR which affect the converging and stabilizing time of FVPATA. (2) The size of the steps in power increase. Smaller steps lower the converging time while bigger steps often overshoot the node degree and waste energy. (3) The limits on total number of degrees. A larger number of degrees results in a higher throughput with more energy consumption. (4) The position of the attacker and the pattern of network flows. Figure 10 considers the position of attackers simulated by matlab and the results indicate that the attacker's position can also effect network connectivity. Network performance deteriorates with the number of affected nodes if the affected nodes are selected by the indication of their Fiedler values. However, FVPATA works well under the condition where multiple affected nodes since the nodes near broken nodes have more opportunity to be selected as Fiedler nodes and thus, increase their power. (5) RTS/CTS, this is a solution to the problems of hidden terminal and the reduction of flow interference. Since it affects the traffic pattern, it also influences the network throughputs. (6) The depth of exploring process in Algorithm 1. A more in-depth exploration of

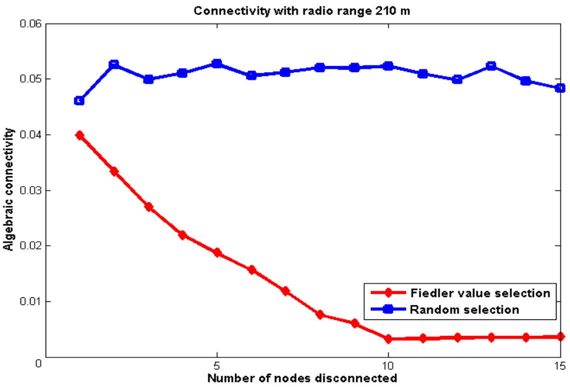


Fig. 10 The relationship between attackers' positions and graph connectivity in a one-hundred-node network.

network connectivity indicates a lesser number of hops in a route, which translates to a much stronger and more robust network. On the other hand, this also causes an overshoot in the node power.

5 Conclusions

In order to achieve topology adaptation for the resilient communication operation of cyberphysical networks deployed using ad hoc networking technologies, we proposed an algorithm called FVPATA to overcome attacks that puncture a hole in the ad hoc network. The objective of FVPATA is to use the Fiedler value, which approximates the robustness of a graph, to guide the nodes in the cyberphysical network to discover weak vertices in the underlying network topology. Each node verifies and promotes itself to be a frail node if it is the Fiedler node. Fiedler nodes increase their transmission power after finding they are weak nodes. FVPATA is totally distributed since each node can obtain self-evaluated topology information through OLSR routing messages, which requires no extra communication messages. After FVPATA converges, the robustness of the network locally around the Fiedler node is enhanced. Moreover, we proposed a method to solve the problems associated with asymmetric links during the process of increasing power so that nodes will not raise their power indefinitely. The final state for the network is that Fiedler nodes are connected to each other, accompanied by some non-Fiedler nodes participating as bridges among them. Those Fiedler nodes and special non-Fiedler nodes compose a backbone for the network. This structure significantly reduces the number of hops along a route and lowers the latency, yielding a higher network efficiency since only a few nodes increased their power.

References

[1] T. He, J. A. Stankovic, C. Lu, and T. Abdelzaher, Speed: A stateless protocol for real-time communication in sensor networks, in *Distributed Computing Systems, 2003. Proceedings 23rd International Conference on*, IEEE, 2003, pp. 46–55.

[2] W. Su and M. Gerla, Ipv6 flow handoff in ad hoc wireless networks using mobility prediction, in *Global Telecommunications Conference, 1999. GLOBECOM99*, IEEE, 1999, vol. 1, pp. 271–275.

[3] C.-K. Toh, Associativity-based routing for ad hoc mobile networks, *Wireless Personal Communications*, vol. 4, no. 2, pp. 103–139, 1997.

[4] S. Marti, T. J. Giuli, K. Lai, and M. Baker, Mitigating routing misbehavior in mobile ad hoc networks, in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, ACM, 2000, pp. 255–265.

[5] Y. Liu and W. Trappe, Localization in peer to peer networks based on q-learning, in *the 40th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, Brisbane, Australia, 2015, to be appeared.

[6] G. G. Helmer, J. S. Wong, V. Honavar, and L. Miller, Intelligent agents for intrusion detection, in *Information Technology Conference*, IEEE, 1998, pp. 121–124.

[7] Y.-A. Huang and W. Lee, A cooperative intrusion detection system for ad hoc networks, in *Proceedings of the 1st ACM Workshop on Security of ad hoc and Sensor Networks*, ACM, 2003, pp. 135–147.

[8] L. Zhou and Z. J. Haas, Securing ad hoc networks, *Network, IEEE*, vol. 13, no. 6, pp. 24–30, 1999.

[9] M. K. Marina and S. R. Das, On-demand multipath distance vector routing in ad hoc networks, in *Network Protocols, 2001. Ninth International Conference on,* IEEE, 2001, pp. 14–23.

[10] M. Al-Shurman, S.-M. Yoo, and S. Park, Black hole attack in mobile ad hoc networks, in *Proceedings of the 42^{nd} Annual Southeast Regional Conference*, ACM, 2004, pp. 96–97.

[11] P. Tague, S. Nabar, J. A. Ritcey, and R. Poovendran, Jamming-aware traffic allocation for multiple-path routing using portfolio selection, *Networking, IEEE/ACM Transactions on*, vol. 19, no. 1, pp. 184–194, 2011.

[12] P. Tague, S. Nabar, J. A. Ritcey, D. Slater, and R. Poovendran, Throughput optimization for multipath unicast routing under probabilistic jamming, in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, IEEE, 2008, pp. 15.

[13] K. Ma, Y. Zhang, and W. Trappe, Mobile network management and robust spatial retreats via network dynamics, in *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, IEEE, 2005, p. 8.

[14] W. Xu, K. Ma, W. Trappe, and Y. Zhang, Jamming sensor networks: Attack and defense strategies, *Network, IEEE* vol. 20, no. 3, pp. 41–47, 2006.

[15] K. Panyim and P. Krishnamurthy, A hybrid key predistribution scheme for sensor networks employing spatial retreats to cope with jamming attacks, *Mobile Networks and Applications*, vol. 17, no. 3, pp. 327–341, 2012.

[16] G. Yue, Anti-jamming coding techniques, *Signal Processing Magazine, IEEE*, vol. 25, no. 6, pp. 35–45, 2008.

[17] G. Yue and X. Wang, Anti-jamming coding techniques with application to cognitive radio, *Wireless Communications, IEEE Transactions on*, vol. 8, no. 12, pp. 5996–6007, 2009.

[18] X. Xu, B. Zheng, J. Zhang, and J. Yan, An efficient anti-jamming piecewise coding in cognitive of dm, in *Communication Technology (ICCT), 2010 12th IEEE International Conference on*, IEEE, 2010, p. 14.

[19] K. Cheun and W. E. Stark, Performance of robust metrics with convolutional coding and diversity in fhss systems under partial-band noise jamming, *Communications, IEEE Transactions on*, vol. 41, no. 1, pp. 200–209, 1993.

[20] O. Sidek and A. Yahya, Reed solomon coding for frequency hopping spread spectrum in jamming environment, *American Journal of Applied Sciences*, vol. 5, no. 10, p. 1281, 2008.

[21] M. Fiedler, Algebraic connectivity of graphs, *Czechoslovak Mathematical Journal*, vol. 23, no. 2, pp. 298–305, 1973.

[22] A. Jamakovic and S. Uhlig, On the relationship between the algebraic connectivity and graphs robustness to node and link failures, in *Next Generation Internet Networks, 3rd EuroNGI Conference on*, IEEE, 2007, pp. 96–102.

[23] N. M. M. de Abreu, Old and new results on algebraic connectivity of graphs, *Linear Algebra and Its Applications*, vol. 423, no. 1, pp. 53–73, 2007.

[24] D. B. Johnson, D. A. Maltz, and J. Broch, Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks, in *Ad Hoc Networking*. Boston, MA, USA: Addison-Wesley Longman Publishing Co. Inc., 2001, pp. 139–172.

[25] C. E. Perkins and E. M. Royer, Ad-hoc on-demand distance vector routing, in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA99. Second IEEE Workshop on*, IEEE, 1999, pp. 90–100.

[26] T.-W. Chen and M. Gerla, Global state routing: A new routing scheme for ad-hoc wireless networks, in *Communications, 1998. ICC 98. Conference Record. 1998 IEEE International Conference on*, IEEE, 1998, vol. 1, pp. 171–175.

[27] J. N. Al-Karaki and A. E. Kamal, Efficient virtualbackbone routing in mobile ad hoc networks, *Computer Networks*, vol. 52, no. 2, pp. 327–350, 2008.

[28] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized link state routing protocol (OLSR), RFC 3626, Internet Engineering, Task Force (IETF), 2003.

[29] S. Jain and S. Sahu, Topology vs position based routing protocols in mobile ad hoc networks: A survey, *International Journal of Engineering Research and Technology*, vol. 1, no. 3, pp. 1–11, 2012.

[30] M. E. Newman, The structure and function of complex networks, *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[31] C. Pandana and K. R. Liu, Robust connectivity-aware energy-efficient routing for wireless sensor networks, *Wireless Communications, IEEE Transactions on*, vol. 7, no. 10, pp. 3904–3916, 2008.

[32] B. Mohar and Y. Alavi, The Laplacian spectrum of graphs, *Graph Theory, Combinatorics, and Applications*, vol. 2, pp. 871–898, 1991.

Wade Trappe is an associate director with the Wireless Information Network Laboratory (WINLAB) and a professor with the Department of Electrical and Computer Engineering at Rutgers University, New Jersey. His research interests include wireless security, wireless networking, and network security. He has led projects that involve security and privacy for sensor networks, physical-layer security for wireless systems, a security framework for cognitive radios, the development of wireless testbed resources, and new radio frequency identification technologies. He has published more than 150 papers, including five best paper awards. He is a co-author of the popular textbook *Introduction to Cryptography With Coding Theory*, and an IEEE Fellow.

Ying Liu is currently a PhD student in WINLAB at Rutgers University, New Brunswick, New Jersey. She received her MS degree in electrical engineering from Shanghai Jiao Tong University, China, in 2008. Her current research focuses on the security of ad hoc networks, with a focus on robust network topologies, identification of adversarial attacks by machine learning, game theory for optimal resource allocation and game theory for modeling security issues in wireless networks, game theory for optimal resource allocation, and game theory for modeling security issues in wireless networks.