Towards Secure Link Quality Measurement in Multihop Wireless Networks

Kai Zeng[†], Shucheng Yu[†], Kui Ren[‡], Wenjing Lou[†], and Yanchao Zhang[*]
[†]Department of ECE, Worcester Polytechnic Institute, MA 01609
{kzeng, yscheng, wjlou}@wpi.edu
[‡]Department of ECE, Illinois Institute of Technology, IL 60616
kren@ece.iit.edu
[*]Department of ECE, New Jersey Institute of Technology, NJ 07102
yczhang@njit.edu

***Abstract*— Link quality measurement (LQM), i.e. packet reception ratio (PRR) measurement, is becoming an indispensable component in multihop wireless networks. However, in all the existing LQM mechanisms, a common fact is that a node's knowledge about the forward PRR from itself to its neighbor is informed by the neighbor. On the one hand, this receiver-dependent measurement provides accurate and timely updates on the link quality. On the other hand, it opens up a door for a malicious node to easily report a false measurement result to mislead the routing decision and degrade the system performance. In this paper, we analyze the security vulnerabilities in the existing LQM mechanisms and propose an efficient broadcast-based secure LQM (SLQM) mechanism, which prevents the malicious receiver from reporting a higher PRR than the actual one. We analyze the security strength and the cost of the proposed mechanism. Simulation results show that even when there are only 10% malicious nodes in the network, the average end-to-end throughput can be degraded by 50% compared with the normally operated network, which demonstrates the importance of employing SLQM mechanisms. To the best of our knowledge, this is the first work addressing the SLQM problem in multihop wireless networks.**

I. INTRODUCTION

The promise of multihop wireless networks to solve challenging real-world problems continues to attract attention from both industry and academia in the past decade. Various crucial applications of multihop wireless networks include emergency response operations, military battle-field communication, last-mile broadband internet access, animal habitat monitoring and tracking, etc. Typically, multihop wireless networks, such as sensor networks and mesh networks, are deployed in large and heterogeneous areas using open wireless media. In such environment, wireless links are highly unreliable and usually experience significant quality fluctuations [1], [2] and present asymmetry [3].

The packet reception ratio (PRR) has been widely used as an indicator of the link reliability in multihop wireless networks. It has been shown that routing performance is significantly improved by considering the link PRR information. For example, *expected transmission count* (ETX) based routing achieves much higher throughput than traditional minimum-hop routing protocols in wireless mesh networks [1]. The link ETX is defined as $\frac{1}{p_f \cdot p_r}$, where p_f and p_r is the forward and reverse link PRR, respectively. Recent work in sensor networks [3] suggests a link metric, *expected number of transmissions over forward links* (ETF), which only considers forward link PRR. State-of-the-art geographic routing protocols [4], [5] and most opportunistic routing protocols [6]–[8] also rely on link quality information to make routing decision.

Providing accurate link quality measurement (LQM) [1] is essential to ensure right operation of the above protocols/schemes. Furthermore, LQM is also important to supporting QoS guarantee in multihop wireless networks. Lastly, accurate long-term statistics of link-quality information is necessary to diagnose a network to identify the source of network failures, and reduce the management overhead.

The existing LQM mechanisms proposed in the literature [1], [3], [9] can be generally classified into three types: active, passive, and cooperative probings [9]. For broadcast-based active probing [1], each node periodically broadcasts hello/probing packets, and its neighbors record the number of received packets to calculate the PRRs from the node to themselves. In passive probing [9], the real traffic generated in the network is used as probing packets without introducing extra overhead. For cooperative probing [9], a node estimates the link quality from its neighbor to itself by overhearing the transmissions of its neighbor.

However, for any of the existing LQM mechanisms, the inherent common fact is that a node's knowledge about the forward PRR from itself to its neighbor is informed by the neighbor. Since multihop wireless networks are generally deployed in an ad hoc style or in untrusted environments, nodes may be compromised and act maliciously. This receiver-dependent measurement opens up a door for malicious attackers to report a false measurement result to disturb the routing decision for all the PRR-based protocols. For example, in Fig. 1, suppose A is the source and D is the destination, and the actual PRR is indicated above each link in Fig. 1(a). The ETF-based shortest path routing would select the path $A \rightarrow B \rightarrow D$, since it has the lowest ETF path cost. However,

[1] In this paper, we mainly focus on PRR measurement. Without specification, the link quality indicates PRR.

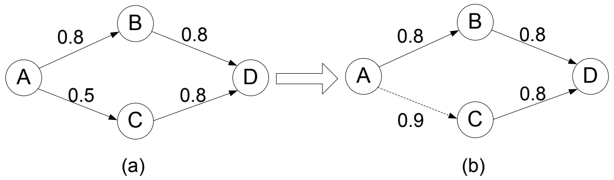


Fig. 1. A 4-node example. (a) The actual PRR on each link is indicated, and the ETF-based routing selects the optimal path $A \rightarrow B \rightarrow D$. (b) The malicious node C bluffs A into believing that the PRR from A to C is 0.9, then the ETF-based routing would select the suboptimal path $A \rightarrow C \rightarrow D$.

if C is a malicious node, and reports to A that the PRR from A to itself is 0.9 (indicated below the link in Fig. 1(b)), then A would select path $A \rightarrow C \rightarrow D$. In such a way, a suboptimal path is selected between A and D, thus degrades routing performance. More severely, C attracts all the traffic from A, then with the control of the traffic, it can further maliciously drop or corrupt the packets.

To the best of our knowledge, none of the existing work addresses security vulnerabilities in the existing LQM mechanisms. As LQM is becoming an indispensable component in multihop wireless networks, it is necessary to make this component work securely and provide actual PRR information for routing protocols and other applications.

In this paper, we analyze the security vulnerabilities in the existing LQM mechanisms. We then propose a broadcast-based secure LQM mechanism, which prevents the malicious attacker from reporting a higher PRR than the actual one. This framework can be easily applied to unicast-based and cooperative LQM mechanisms. Simulation results show that the average end-to-end throughput can be severely degraded even when there is only a small portion of malicious nodes in the network, which demonstrates the importance of employing SLQM mechanisms in multihop wireless networks.

The rest of this paper is organized as follows. Section II introduces the existing link quality measurement mechanisms and point out their security pitfalls. We propose a broadcast-based secure LQM (SLQM) mechanism and analyze its security strength and overhead in Section III. Simulation results are presented in Section IV. Conclusions are drawn in Section V.

II. Existing Link Quality Measurement Mechanisms and Vulnerabilities

This section gives an overview of the existing LQM mechanisms and analyzes their security vulnerabilities. According to the type of probing packets, LQM can be classified into broadcast-based and unicast-based probing. While based on the generation source of probing packets, LQM can also be categorized into active, passive, and cooperative probing [9].

A. Broadcast-based Active Probing

For broadcast-based active probing [1], each node broadcasts link probes of a fixed size, at an average period τ (e.g. 1 second). Every node remembers the probes it receives during the last w seconds (e.g. 10 seconds), allowing it to calculate the PRR from the measuring node at any time t as: $r(t) = \frac{count(t-w,t)}{w/\tau}$, where $count(t-w,t)$ is the number of probes received during the window w, and w/τ is the number of probes that should have been received. In the case of two neighboring nodes A and B, this technique allows A to measure the PRR from B to A, and B to measure the PRR from A to B. Each probe sent by a node A contains the number of probing packets received by A from each of its neighbors during the last w seconds. This allows each neighbor of A to calculate the forward link PRR to A whenever it receives a probe from A.

The security vulnerability in the broadcast-based active probing is that a malicious node can easily report a false measurement result. For example, if node B is an attacker, it can bluff A into believing that the PRR from A to itself is 1 by claiming that it received w/τ packets in the last probing window w.

B. Unicast-based Passive Probing

Unicast-based passive probing [9] makes use of the real unicast traffic as the “natural” probing packets without incurring extra overhead. It is applicable when there is enough unicast traffic on a measured unidirectional link. It runs as follows: for instance, suppose node A has enough traffic to node B. Then, A gets the information about the number of successful transmissions (N_s) and the total number of transmissions (N_t) from its MAC’s MIB (Management Information Base) for the traffic. At the end of an update period, the PRR is derived as $\frac{N_s}{N_t}$, and is further smoothed by moving average [9].

For unicast-based passive probing, it is hard but not impossible for an attacker to cheat on the link quality. In 802.11 [10], the Distributed Coordination Function (DCF) defines two access mechanisms for packet transmissions: basic access mechanism, and RTS/CTS access mechanism. We analyze the security vulnerability of the unicast-based passive probing under these two access mechanisms as follows.

In the basic access mechanism, a sender starts the transmission of a DATA frame after it senses the channel is idle for a while. Upon successful decoding the whole DATA frame, the receiver sends an ACK frame back to the sender, indicating successful reception of the DATA frame. In this case, even when it can not decode the whole data frame, a receiver may decode some parts of it [11]. So it is possible for a malicious receiver to figure out the sender’s address and send back an ACK to claim a correct reception even when it receives a corrupted data frame.

The RTS/CTS access mechanism uses a four-way handshake in order to reduce bandwidth loss due to the hidden terminal problem. Different from the basic access mechanism, a sender will send a RTS frame to the receiver before it sends out the DATA frame. Upon successful reception of the RTS frame, the receiver then sends a CTS frame back to the sender. The sender can start sending the DATA frame after the reception of the CTS frame. As in the basic access mechanism, upon successful reception of the DATA frame, the receiver sends an ACK frame back to the sender. In this case, by receiving the RTS, a malicious receiver can figure out the sender’s address, so even it receives a corrupted data frame, it can still claim a successful reception by sending back an ACK.

In summary, although a sender estimates the link quality based on its own MIB information in the unicast-based passive probing, this information is still dependent on the feedback (ACK) from the receiver. A malicious receiver may still be able to make use of the ACK to bluff the sender into believing that there exists a high quality link from the sender to the receiver.

C. Cooperative Probing

Cooperative probing [9] is used when there is not enough unicast traffic from a measuring node to its neighbor, but to others. For example, a measuring node A has two one-hop neighbors, B and C. A has no egress traffic to C, but to B. The neighbor node (C) with no traffic to it from the measuring node (A) is called a "cooperative" node. Due to the broadcast nature of wireless media, the node C can overhear the traffic from the measuring node A to B. This traffic is called *cross traffic*. The overhearing result is then used for the measuring node to derive the quality of link $A \rightarrow C$. [9] assumes node C cannot receive duplicate frames from its MAC layer even in the promiscuous mode, the retransmitted packets are not used for measurements. So node A counts first-time successful transmissions (C_c) within the cross traffic. In the update period, a report of overheard results (C_a) from C is sent to A, and then the PRR in this period is calculated as $\frac{C_a}{C_c}$.

To attack cooperative probing, similar to the unicast-based passive probing, a malicious "cooperative" node does not need to decode the whole data frame correctly. As long as it can figure out the sender's address and the status (0/1) of the "retry" bit in the data frame, it can increase its count of C_a.

D. Unicast-based Active Probing

When there is no egress/cross traffic, unicast-based active probing can be applied [9]. For example, if node A has no traffic to B or C, A initiates a unicast-based active probing on link $A \rightarrow B$ by generating unicast probing packets. Then, the link quality from A to B is measured in the same way as passive probing. At the same time, the quality of link $A \rightarrow C$ can be measured by cooperative probing. In this way, unicast-based active probing acts similarly as the broadcast-based active probing, with difference being in that in unicast-based probing the receiver needs to send back an ACK to the sender when it receives the data frame correctly and the sender will retransmit data frames when no ACK has been received. While in broadcast-based active probing, no node needs to send ACK.

For unicast-based active probing, the security vulnerabilities in measuring the link quality from the measuring node (e.g. A) to the intended receiver (e.g. B) and to the "cooperative" node (e.g. C) are the same as those in unicast-based passive probing and "cooperative" probing, respectively.

To sum up, all the existing LQM mechanisms can not prevent a receiver cheating on the PRR. The inherent fact is that the receiver can claim a correct data frame reception without showing any evidence. To fix this vulnerability, we propose a broadcast-based secure LQM (SLQM) mechanism in the following section. We will show that this broadcast-based mechanism can be easily applied to unicast-based and cooperative SLQM mechanisms.

III. Broadcast-based Secure Link Quality Measurement

In this section, we propose a broadcast-based secure LQM mechanism, and then analyze its security strength and its computation, storage, and communication overhead. In this paper, we assume that a malicious node always wants to report a higher PRR than the actual measured one to disturb PRR-based routing protocols. We also assume that a unique pair-wise key has been established between each pair of neighbors. The neighborhood pair-wise key establishment mechanisms have been extensively studied in multihop wireless networks since [12].

A. Broadcast-based SLQM Framework

Assume a node A has N one-hop neighbors $A_1, A_2, ..., A_N$, and needs to measure the link PRR (p_i) to each of its neighbors (A_i). Similar to [9], the measurement is done periodically. Each measurement period consists of three consecutive phases: probing, reporting, and updating phases, which are described as follows.

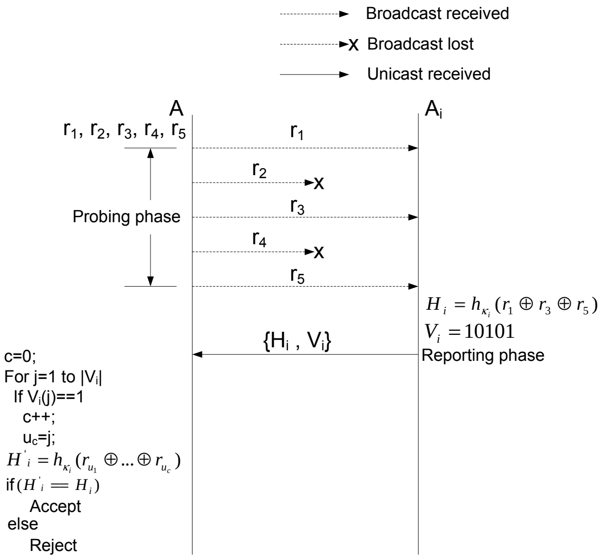
Probing phase: In this phase, A broadcasts N_s packets to its neighbors. The j^{th} packet r_j embeds a random number. Node A keeps the broadcasted packets in its buffer within this measurement period. Receiver A_i only stores the XOR-ed result (R_i) of all the correctly received packets, and the corresponding indicator vector V_i defined in Eq. (1) that indicates the index of the received packet. Note that A_i can compute the XOR-ed result on the fly whenever it receives a new probing packet.

$$V_i(j) = \begin{cases} 1, & A_i \text{ received the } j^{th} \text{ packet correctly;} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

where $V_i(j)$ is the j^{th} bit from the higher (left) end of the vector V_i.

Reporting phase: When the probing phase is ended, each neighbor A_i sends A a report $Rep_i := \{H_i, V_i\}$, where $H_i = h_{\mathcal{K}_i}(R_i)$ is a keyed hash of R_i with the pairwise key $\mathcal{K}_i$ shared between A and A_i. The hash function can be any of the existing cryptographic hash functions, such as MD5.

Updating phase: On receiving A_i's report, A figures out how many and which packets A_i received in the probing phase by examining the number and positions of bit '1's in vector V_i. Since A keeps all the packets that it broadcasted, it computes $R_i^{'}$ by doing XOR of the packets that A_i claims it received. A then computes $H_i^{'} = h_{\mathcal{K}_i}(R_i^{'})$. If $H_i^{'} = H_i$, A accepts this report; otherwise, it rejects the report. Suppose A counts there are N_{r_i} bit '1's in V_i, after A accepts the report, A calculates the PRR $p_i = \frac{N_{r_i}}{N_s}$ in this measurement period. A moving average method is further used to smooth the measured result. Denote the measured result in the k^{th} measurement period as $p_i[k]$, the smoothed PRR, $\widetilde{p}_i[k]$, at the end of the k^{th} period is calculated as



```
c=0;
For j=1 to |V_i|
  If V_i(j)==1
    c++;
    u_c=j;
$H'_i = h_{\mathcal{K}_i}(r_{u_1} \oplus ... \oplus r_{u_c})$
if($H'_i == H_i$)
    Accept
else
    Reject
```

Fig. 2. Probing and reporting phases of secure link quality measurement between A and A_i in a measurement period

$$\widetilde{p}_i[k] = (1-\alpha)\widetilde{p}_i[k-1] + \alpha p_i[k] \quad (2)$$

where α is a smoothing constant in the range of (0,1).

Figure 2 shows an example of the broadcast-based SLQM mechanism in a measurement period. Suppose in the probing phase, A broadcasts 5 probing packets (r_1,...,r_5), and A_i receives the packets r_1, r_3, and r_5. In the reporting phase, A_i calculates $H_i = h_{\mathcal{K}_i}(r_1 \oplus r_3 \oplus r_5)$, then sends H_i and a 5-bit vector $V_i = 10101$ back to A. When A receives the H_i and V_i, it examines V_i and gets the indices $(u_1, ..., u_c)$ of the packets A_i claims it has received, then calculates $H_i' = hash_{\mathcal{K}_i}(r_{u_1} \oplus ... \oplus r_{u_c})$. If $H_i = H_i'$, A accepts A_i's report; otherwise, rejects it.

B. Security Strength

We now analyze the security strength of our broadcast-based SLQM mechanism. This mechanism achieves the security goal that prevents a malicious attacker from reporting a higher PRR than the actual one. We assume A_i is malicious in the following discussion.

First, it's computationally impossible for A_i to guess the packets which it does not receive, even when A_i overhears other's report. For example, in Figure 2, if A_i wants to claim it receives r_1, r_3, r_4, r_5, it needs to create a hash value $H_i = h_{\mathcal{K}_i}(r_1 \oplus r_3 \oplus r_4 \oplus r_5)$. Since it has no idea what r_4 is, the only thing it can do is to make a guess on r_4. However, it's hard to make a correct guess according to the weak collision resistance property of the hash function that given $x = (r_1 \oplus r_3 \oplus r_4 \oplus r_5)$, it's hard to find a $y = r_1 \oplus r_3 \oplus r_4^{'} \oplus r_5$, such that $h_{\mathcal{K}_i}(x) = h_{\mathcal{K}_i}(y)$. Even A_i overhears A_j's report indicating that A_j receives r_4, A_i still can not get any information about r_4 because of the one-way property of the hash function.

Second, our mechanism prevents A_i from replaying its own or other neighbor's report. According to the randomness embedded in each probing packet, even A_i receives all the probing packets in some measurement period, it can not replay this report in the following measurement period. Furthermore, if A_i replays A_j's report, this report can not pass the verification by A, because A uses $\mathcal{K}_i$ instead of $\mathcal{K}_j$ to verify A_i's report.

C. Computation, Storage and Communication Overhead

Computation overhead: On the sender side, A needs to generate a random number sequence. According to its computation and storage capability, A can generate a large random number sequence to be used for several measurement periods, and refresh this sequence when it is used up. Any of the existing efficient pseudorandom number generators, such as linear congruential generator [13], can serve this purpose. To do verification, A only needs to do XOR and hash operations, which are computationally efficient. On the receiver side, to create the report digest, each neighbor only needs to do a hash computation.

Storage overhead: On the sender side, A only needs to store the generated random numbers. Suppose the length of each random number is L_r bytes, the probing packet broadcast rate is B packet/second, and the probing phase is P seconds. Then in a measurement period, A needs $S = L_r \cdot B \cdot P$ bytes storage space. For example, if $L_r = 16$, $B = 1$, and $P = 10$, $S = 160bytes$, which is supportable even on sensor nodes.

Communication overhead: The communication overhead of our SLQM mechanism is comparable to any existing broadcast-based probing mechanism, such as that in [1]. As the probing packet broadcast rate is usually low, e.g. $B = 1$, SLQM introduces very light local traffic into the network.

D. Applicability

As discussed above, our SLQM mechanism has very low computation, storage and communication overhead, so it's applicable to resource-constraint networks, such as wireless sensor networks, as well as more powerful networks, such as wireless mesh networks. Basically, broadcast-based SLQM can be implemented at application, network or MAC layer. Our SLQM framework can also be easily applied to unicast-based and cooperative LQM with a slight modification such that we embed a random number in each unicast packet (including retransmitted packets at MAC layer). For unicast-based SLQM, we can ask receiver to attach a hash value of the received packet in the corresponding ACK. For cooperative probing, the cooperative receiver does the same thing as the broadcast-based SLQM.

IV. Performance Demonstration

In this section, we simulate ETF-based shortest path routing [3] in a sensor network scenario to demonstrate the performance degradation of PRR-based routing when some nodes intend to report a false PRR. The simulations are implemented within the GloMoSim simulator [14]. The simulated network has 196 stationary nodes randomly uniformly distributed in a $d \times d$ m^2 square region. We vary d as 330, 250, 210, and 180 to examine the routing performance under different

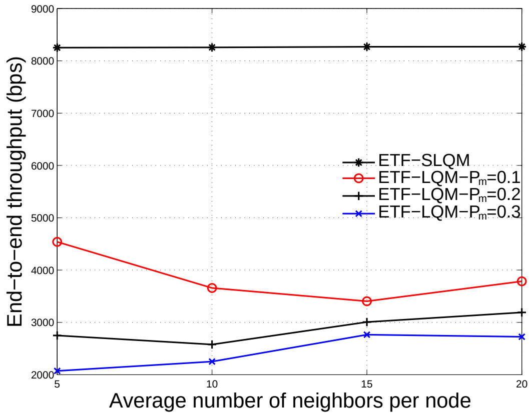


Fig. 3. Average end-to-end throughput under different node densities and malicious node portions.

node densities in terms of $5, 10, 15, 20$ neighbors per node on average. We assume Ground Reflection (Two-Ray) path loss model and Ricean fading model for signal propagation. The broadcast-based LQM is used to measure the PRR on each link, such that each node broadcasts one probing packet per second, and for every 10 seconds (measurement period), each node reports its neighbors the PRR information from them to itself. Each node updates the PRR value to its neighbors according to Eq. (2) when it receives the report from its neighbors. The α in Eq. (2) is chosen to be 0.9. For SLQM, we assume forced by the security mechanism, nodes faithfully report the measured result. For non-secured LQM, we assume malicious nodes will always report a 0.9 PRR no matter what the actual one is. The malicious nodes are randomly distributed in the network. To study the impact of number of malicious nodes on the performance, we vary the portion (P_m) of malicious nodes as 0.1, 0.2 and 0.3. IEEE $802.11b$ [10] is used as the MAC layer protocol. Each node transmits packets at 2Mbps. We randomly choose 15 communication pairs running CBR (constant bit rate) applications. The CBR rate is two packets per second and each packet being 512 bytes long. Each point in the plotted results represents an average of 10 simulation runs with different seeds. The performance metric we examine is the end-to-end throughput, which is the average throughput of the 15 communication pairs.

From Fig. 3, we can see that under all the different node densities, even when there are only 10% malicious nodes in the network, the throughput can be degraded by 50% compared with that in the normally operated network which is secured by our SLQM mechanism. The throughput decreases when the portion of malicious nodes increases. Actually, there are many packets are dropped due to retransmission limit in a malicious environment where attackers bluff their neighbors into believing that there are “good” links from the neighbors to themselves. The simulation results indicate the importance of employing SLQM mechanism which prevents malicious nodes from reporting a false link measurement result.

V. Conclusions and Future Work

In this paper, we investigated the existing link quality measurement mechanisms, and analyzed the security vulnerabilities in them. A common inherent fact in all the existing LQM mechanisms are receiver-dependent measuring, that is, a node’s knowledge about the forward PRR from itself to its neighbors is informed by its neighbors. We then proposed a broadcast-based secure LQM mechanism that prevents a neighboring node from maliciously claiming a higher measurement result. Our mechanism has very low computation, storage, and communication overhead, thus can be implemented in resource-constrained sensor networks as well as mesh networks. Our SLQM mechanism can be easily applied to unicast-based and cooperative LQM with slight modifications. The simulation results demonstrated the importance of employing SLQM mechanisms in multihop wireless networks. As for the future work, we are interested in defending more sophisticated attacks such as collusion among multiple neighbors.

Acknowledgment

Kai Zeng, Shucheng Yu and Wenjing Lou are supported in part by the US National Science Foundation under grants CNS-0626601, CNS-0716306, and NSF CAREER Award CNS-0746977. Kui Ren is supported in part by ERIF, IIT. Yanchao Zhang is supported in part by the US National Science Foundation under grant CNS-0716302.

References

[1] D. Couto, D. Aguayo, J. Bicket, and R. Morris, “A high-throughput path metic for multi-hop wireless routing,” in *ACM MobiCom’03*, San Diego, California, Sept. 2003.

[2] J. Zhao and R. Govindan, “Understanding packet delivery performance in dense wireless sensor networks,” in *ACM Sensys’03*, LA,CA, Nov. 2003.

[3] L. Sang, A. Arora, and H. Zhang, “On exploiting asymmetric wireless links via one-way estimation,” in *MobiHoc ’07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*. New York, NY, USA: ACM, 2007, pp. 11–21.

[4] K. Seada, M. Zuniga, A. Helmy, and B. Krishnamachari, “Energy efficient forwarding strategies for geographic routing in wireless sensor networks,” in *ACM Sensys’04*, Baltimore, MD, Nov. 2004.

[5] K. Zeng, K. Ren, W. Lou, and P. J. Moran, “Energy aware efficient geographic routing in lossy wireless sensor networks with environmental energy supply,” *Wireless Networks (WINET)*, pp. 477–486, 2007.

[6] S. Biswas and R. Morris, “Exor: Opportunistic multi-hop routing for wireless networks,” in *SIGCOMM’05*, Philadelphia, Pennsylvania, Aug. 2005.

[7] K. Zeng, W. Lou, J. Yang, and D. R. Brown, “On throughput efficiency of geographic opportunistic routing in multihop wireless networks,” in *QShine’07*, Vancouver, British Columbia, Canada, August 2007.

[8] K. Zeng, W. Lou, and H. Zhai, “On end-to-end throughput of opportunistic routing in multirate and multihop wireless networks,” in *IEEE Infocom’08*, Phoenix, AZ, April 15-17 2008.

[9] K.-H. Kim and K. G. Shin, “On accurate measurement of link quality in multi-hop wireless mesh networks,” in *MobiCom ’06: Proceedings of the 12th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM Press, 2006, pp. 38–49.

[10] *IEEE Std 802.11b-1999*. [Online]. Available: http://standards.ieee.org/

[11] K. Jamieson and H. Balakrishnan, “Ppr: Partial packet recovery for wireless networks,” in *ACM SIGCOMM*, Kyoto, Japan, August 2007.

[12] L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” in *CCS ’02: Proceedings of the 9th ACM conference on Computer and communications security*. New York, NY, USA: ACM, 2002, pp. 41–47.

[13] M. Greenberger, “Notes on a new pseudo-random number generator,” *J. ACM*, vol. 8, no. 2, pp. 163–167, 1961.

[14] X. Zeng, R. Bagrodia, and M. Gerla, “Glomosim: a library for parallel simulation of large-scale wireless networks,” in *Proceedings of PADS’98*, Banff, Canada, May 1998.