This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2005 proceedings.

Anti-Attack Cooperation Stimulation in Self-organized Ad Hoc Networks

Wei Yu and K. J. Ray Liu
Department of Electrical and Computer Engineering and The Institute for Systems Research
University of Maryland, College Park, MD 20742
Email: weiyu, kjrliu@isr.umd.edu

***Abstract*— In self-organized ad hoc networks, nodes usually belong to different authorities and pursue different goals. To save their valuable resources, they tend to be selfish. Meanwhile, some nodes may be malicious whose objective is to degrade the network performance. In this paper, we present an anti-attack cooperation stimulation system for self-organized ad hoc networks to stimulate cooperation among selfish nodes and to defend against attacks. In the proposed system, the damage that can be caused by attackers is bounded, and the cooperation among selfish nodes is enforced. Simulation studies also verify the effectiveness of the proposed system. Another key property of the proposed system is that it is fully distributive, and does not require any tamper-proof hardware or central management point.**

I. Introduction and Background

An *ad hoc network* is a group of (possible mobile) nodes without requiring centralized administration or fixed network infrastructure, in which nodes can communicate with other nodes out of their direct transmission ranges through cooperatively forwarding packets for each other. Since ad hoc networks can be easily and inexpensively set up as needed, they have a wide range of applications in emergency or military applications. Recently, emerging applications of ad hoc networks are also envisioned in civilian usage [1], where nodes typically do not belong to a single authority and may not pursue a common goal. Consequently, fully cooperative behaviors such as unconditionally forwarding packets for others cannot be directly assumed. On the contrary, in order to save limited resources, nodes may tend to be "selfish". We refer to such networks as self-organized ad hoc networks.

Before ad hoc networks can be successfully deployed in an self-organized way, *cooperation stimulation* and *security* issues must be resolved first. To stimulate cooperation among selfish nodes, one possible way is to apply payment-based approaches, such as [2]–[4]. The drawback of these schemes lies in that they require either tamper-proof hardware or online central management points. Moreover, these schemes have only considered nodes' selfish behaviors. Another possible way to stimulate cooperation is to employ reputation-based schemes [5]–[7]. However, these schemes also suffer from some problems. First, many attacks can cause malicious behavior not being detected. Second, these schemes can only isolate misbehaving nodes, but cannot actually punish them, and the attackers can still utilize the valuable network resources even after they have been suspected or detected.

Previous experiences have also shown that before ad hoc networks can be successfully deployed, security concerns must be addressed [8]–[11]. Meanwhile, past experiences also show that security in ad hoc networks is particularly hard to achieve. For self-organized ad hoc networks, things are even worse: there are no centralized monitoring or management points and nodes may tend to be selfish. In the literature many schemes have been proposed to address the security issues in ad hoc networks. However, most of the focus is on preventing attackers from entering the network through secure key distribution and secure neighbor discovery, such as [10]–[12], they cannot handle well the situation that the attackers have entered the network, while in self-organized ad hoc networks the access control is usually loose, and malicious users can easily join the network.

In this paper we consider the scenarios that there exist both selfish and malicious nodes in self-organized ad hoc networks. The objective of selfish nodes is to maximize the benefits they can get from the network, while the objective of attackers is to maximize the damage they can cause to the network. In this paper, we propose an anti-attack cooperation stimulation system for self-organized ad hoc networks to stimulate cooperation among selfish nodes in adversarial environments. Besides being robust to various attacks, another key property of the proposed system is that it does not require any tamper-proof hardware or central management point. Both system analysis and simulation studies will confirm the effectiveness of the proposed system.

The rest of the paper is organized as follows. Section II describes the system model and formulates the problem. Section III describes the proposed system and analyze the performance under attacks. Simulation studies are presented in Section IV. Finally, Section V concludes this paper.

II. System Model and Problem Formulation

In this paper we consider self-organized ad hoc networks where nodes belong to different authorities and have different goals. We assume that each node is equipped with a battery with limited power supply, and may act as a service provider: packets are scheduled to be generated and delivered to certain destinations with each packet having a specific delay constraint. If a packet can be successfully delivered to its

This work was supported in part by the Army Research Office under URI Award No. DAAD19-01-1-0494

TABLE I

$C_{redit}(A,S)$	Total energy that A has spent until the current moment on successfully transmitting packets for S.
$D_{ebit}(A,S)$	Total energy that S has spent until the current moment on successfully transmitting packets for A.
$W_{by}(A,S)$	Total wasted energy that A has caused to S until the current moment.
$W_{to}(A,S)$	Total wasted energy that S has caused to A until the current moment.
$LB_{with}(A,S)$	Total wasted energy caused to S until the current moment due to link breaks between A and S.
$B_{lacklist}(A,S)$	The subset of A's blacklist known by S until the current moment.

destination within the specified delay constraint, the source of the packet S will get some payoff α_S, otherwise, it will be penalized β_S. According to their objectives, the nodes in the network can be classified into two types: selfish and malicious.

In this paper we mainly consider the situations that all nodes in the network are legitimate, no matter selfish or malicious. We assume that each node has a public/private key pair. We also assume that a node can authenticate the other nodes' public keys, but no node will disclose its private key to the others unless it has been compromised. We assume that the *acknowledgement* mechanism is supported in the link layer. We also assume that all data packets have the same size, and the transmitting power is fixed for all nodes.

Let E_S be the amount of energy that S has been spent. For each selfish node S, let $E_{S,max}$ be its total available energy when it enters the network, and E_S be the amount of energy that it has been spent, then its objective is formulated as:

$$\max\left(\alpha_S N_{S,succ} - \beta_S N_{S,fail}\right), \quad \text{s.t.} \quad E_S \le E_{S,max}, \tag{1}$$

where $N_{S,succ}$ and $N_{S,fail}$ are the total number of data packets that have been successfully and unsuccessfully delivered with S being the source, respectively.

If S is malicious, let $E_{S,waste}$ be the amount of selfish nodes' energy that has been wasted by S due to its malicious behavior, and $E_{S,contr}$ be the amount of energy that S has spent to successfully forward packets for the selfish nodes, then the total damage D_S that S has caused to the selfish nodes can be calculated as

$$D_S = E_{S,waste} - E_{S,contr}. \tag{2}$$

Since in the current system model attackers are allowed to collude, the attackers' overall objective of attackers can be formulated as follows:

$$\max \sum_{\text{S is malicious}} D_S. \tag{3}$$

III. System Description

This section presents the proposed system for self-organized ad hoc networks. In the proposed system, each selfish node S keeps a set of records indicating the interactions with the other nodes, as listed in Table I.

A. Cooperation Degree

In the proposed system, each selfish node S keeps track of the *balance* $B(A,S)$ with any other node A known by S, which is defined as:

$$B(A,S) = (D_{ebit}(A,S) - W_{to}(A,S)) - (C_{redit}(A,S) - W_{by}(A,S)). \tag{4}$$

That is, $B(A,S)$ is the difference between what S has contributed to A and what A has contributed to S in S's point of view. If $B(A,S)$ is a positive value, it can be viewed as the relative damage that A has caused to S, otherwise, it is the relative help that S has received from A.

Besides keeping track of the balance, each node S also sets a threshold $B_{th}(A,S)$ for each known node A in the network, which we called *cooperation degree*. A necessary condition for S to forward packet for A is that

$$B(A,S) < B_{th}(A,S). \tag{5}$$

Setting $B_{th}(A,S)$ to be ∞ means S will always help A no matter what A has done. Setting $B_{th}(A,S)$ to be $-\infty$ means S will never help A. In the proposed system, each selfish node will set $B_{th}(A,S)$ to be a relatively small positive value, which means that initially S is helpful to A, and will keep being helpful to A unless the relative damage that A has caused to S exceeds $B_{th}(A,S)$. By specifying positive cooperation degrees, cooperation among selfish nodes can be enforced, while by letting the cooperation degrees to be relatively small, the possible damage that can be caused by attackers can be bounded.

B. Route Selection

Assume that S has a packet scheduled to be sent to D, route $R =$ "$R_0 R_1 \ldots R_M$" is a valid route known by S with $R_0 = S$, $R_M = D$, and M being the number of hops. Let $P_{drop}(R_i,S)$ be the probability that node R_i will drop S's packet, and let $P_{delivery}(R,S)$ denote the probability that a packet can be successfully delivered from S to D through route R at the current moment which can be calculated as follows:

$$P_{delivery}(R,S) = \begin{cases} 0 & (\exists R_i \in R)\ B(R_i,S) < -B_{th}(S,R_i) \\ 0 & (\exists R_i, R_j \in R)\ R_i \in B_{lacklist}(R_j,S) \\ \prod_{i=1}^{M-1}(1 - P_{drop}(R_i,S)) & otherwise \end{cases} \tag{6}$$

That is, a packet delivery has no chance to succeed unless S has enough balance to request help from all intermediate nodes on the route and no node has been marked as malicious by any other node on the route. Let E be the amount of energy needed to transmit one packet. Once a valid route R with non-zero $P_{delivery}(R,S)$ is used to send a packet by S, the expected energy consumption of using route R to send a packet from S to D, $E_{avg}(R,S)$, becomes

$$MEP_{delivery}(R,S) + \sum_{n=1}^{M-1} nE\Big(\prod_{k=1}^{n-1}(1 - P_{drop}(R_k,S))\Big)P_{drop}(R_n,S) \tag{7}$$

and the expected profit of S is

$$P_{rofit}(R,S) = \alpha_S P_{delivery}(R,S) - \beta_S(1 - P_{delivery}(R,S)). \tag{8}$$

Let $Q(R,S)$ be the expected profit per unit energy when S uses R to send a packet to D at the current moment, namely expected *energy efficiency*, which is the ratio of $P_{rofit}(R,S)$

over $E_{avg}(R,S)$. In the proposed system, the following criterion is used when a node makes route selection decisions: *Among all routes $\mathcal{R}$ known by S which can reach D, route R^* will be selected if and only if $P_{delivery}(R^*,S) > 0$ and $Q(R^*,S) \geq Q(R,S)$ for any $R \in \mathcal{R}$.*

C. Data Packet Delivery Protocol

In the proposed system, the data packet delivery consists of two stages: *forwarding data packet stage* and *submitting receipts stage*. In the forwarding data packet stage, the data packet is delivered from the source to the destination, while in the submitting receipts stage, each participating node on the route tries to submit a receipt to the source to claim credit.

1) Forwarding Data Packet Stage: Suppose that node S is to send a packet with payload m and sequence number $seq_S(S,D)$ to destination D through the route R. For the sender S, it first computes a signature $s = sign_S(MD(m), R, seq_S(S,D))$. Next, S transmits the packet $(m, R, seq_S(S,D), s)$ to the next node on the route, increases $seq_S(S,D)$ by 1, and waits for receipts to be returned by the following nodes on route R. Once a selfish node A has received the packet $(m, R, seq_S(S,D), s)$, A first checks whether itself is the destination of the packet. If it is the destination, after verifications, A returns a receipt to its previous node on the route to confirm the successful delivery; otherwise, A checks whether this packet should be forwarded. A is willing to forward the packet if and only if all the following conditions are satisfied: 1) A is on the route R; 2) $seq_S(S,D) > seq_A(S,D)$, where $seq_A(S,D)$ is the sequence number of the last packet that A has forwarded with S being source and D being the destination; 3) the signature is valid; 4) $B(S,A) < B_{th}(S,A)$; 5) no node on route R has been marked as malicious by A.

Once A has successfully forwarded $(m, R, seq_S(S,D), s)$ to the next node on the route, it will specify a time to wait for a receipt being returned by the next node to confirm the successful transmission, which A will use to claim credit from S. In the proposed system, a selfish node sets its waiting time to be the value of T_{link} multiplied by the number of hops following this node, where T_{link} is a relatively small interval to account for the necessary processing and waiting time per hop. Since in general the waiting time is small enough, we can assume that if a node can return a receipt to its previous node in time, the two nodes will still keep directly connected.

2) Submitting Receipts Stage: After a node (e.g. A) has forwarded a packet $(m, R, seq_S(S,D), s)$ for another node (e.g. S), A will try to claim corresponding credits from S which A can use later to request S to return the favor. To claim credits from S, A needs to submit necessary evidence to convince S that it has successfully forwarded packets for S. In the proposed system, in order for A to show that it has successfully forwarded a packet for S, A only needs to request a receipt from its next node on the route (e.g. B) indicating that B has successfully received the packet. One possible format of such a receipt can be $\{MD(m), R, seq_S(S,D), B, sign_B(MD(m), R, seq_S(S,D), B)\}$.

Actually, a receipt generated by any node following A on the route can be used as the evidence to convince S that A has successfully forwarded a packet for S. Sometimes a packet will be dropped due to link breakage or the requester running out of balance, and sometimes the next node may not return a receipt even if the transmission is successful, such as when the next node is malicious. For each selfish node, if it has dropped the packet or cannot get a receipt from its next node in time, or the received receipt is not valid, it will generate a receipt by itself and return it to its previous node, otherwise, it will simply send the received receipt back to its previous node on the route.

D. Update Records

In the proposed system, after a packet delivery transaction has finished, no matter whether it is successful or not, each participating node will update its records to keep track of the changing relationships with other nodes and to detect possible malicious behavior. Next we use Fig. 1 to illustrate the records updating procedure, where S is the initiator of the transaction, D is the destination, and "$R = S \ldots AMB \ldots D$" is the associated route.

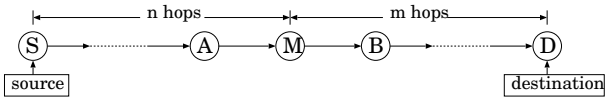


Fig. 1. Records updating

For the sender S, according to different situations, it updates its records as follows:

- *Case 1*: S has received a valid receipt signed by D which means that this transaction has succeeded. In this case, for each intermediate node X between S and D, S increases $C_{redit}(X,S)$ by E.
- *Case 2*: S has successfully sent a packet to its next node, but cannot receive any receipt in time. In this case, let X be S's next node, S then increases $W_{by}(X,S)$ by E, and marks X as malicious. That is, refusing to return a receipt will be regarded as malicious behavior.
- *Case 3*: If S has received a valid receipt which is not signed by D, but signed by an intermediate node (e.g. M), which means either M has dropped the packet, or a returned receipt has been dropped by a certain node following M (including M) on the route in the submitting receipt stage. In this case, for each intermediate node X between S and M, S still increases $C_{redit}(X,S)$ by E. Since node M's transmission cannot be verified by S, S has enough evidence to suspect that the packet is dropped by M. To reflect this suspect, S increases $W_{by}(M,S)$ by nE to account for the amount of energy that has been wasted in this transaction with n being the number of hops between S and M.

For each intermediate node (e.g., node M in Fig. 1) that has participated in the transaction, if it is selfish, it updates its records as follows:

- *Case 1*: M has successfully sent the packet to node B, and has got a receipt from B to confirm the transmission. Then M only needs to increases $D_{ebit}(S, M)$ by E.
- *Case 2*: M has successfully sent the packet to node B, but cannot get a valid receipt from B. In this case, M increases $W_{to}(S, M)$ by nE, increases $W_{by}(B, M)$ by $(n+1)E$, and marks B as malicious.
- *Case 3*: M has dropped the packet due to link breakage between M and B. Although this packet dropping is not M's fault, since M cannot prove it to S, M will take the responsibility. However, since this link breakage may be caused by S who has selected a bad route, or caused by B who tries to emulate link breakage to attack M, M should also records this link breakage. In this case, M increases $W_{to}(S, M)$ by nE, increases $LB_{with}(B, M)$ and $LB_{with}(S, M)$ by nE. In the proposed system, each selfish node (e.g. M) sets a threshold $LB_{th}(S, M)$ with any other node (e.g. S) to indicate the damage that M can tolerate which is caused due to link breakages between M and S. In this case, if $LB_{with}(B, M)$ exceeds $LB_{th}(B, M)$, B will be put in M's blacklist. Similarly, if $LB_{with}(S, M)$ exceeds $LB_{th}(S, M)$, S will be put in M's blacklist.
- *Case 4*: M has dropped the packet due to the reason that the condition in (5) is not satisfied or some nodes on R are in M's blacklist. In this case M does not need to update its records.

From the above update procedure we can see that a selfish node will always return a receipt to confirm a successful packet reception, since refusing to return receipt is regarded as malicious behavior and cannot provide any gain.

E. Secure Route Discovery

In the proposed system, DSR [13] is used as the underlying routing protocol to perform route discovery. However, without security consideration, the routing protocol can easily become an attacking target. For example, attackers can inject an overwhelming amount of route request packets to overload the network and consume other nodes' valuable resource. In the proposed system, the following security enhancements have also been incorporated into DSR:

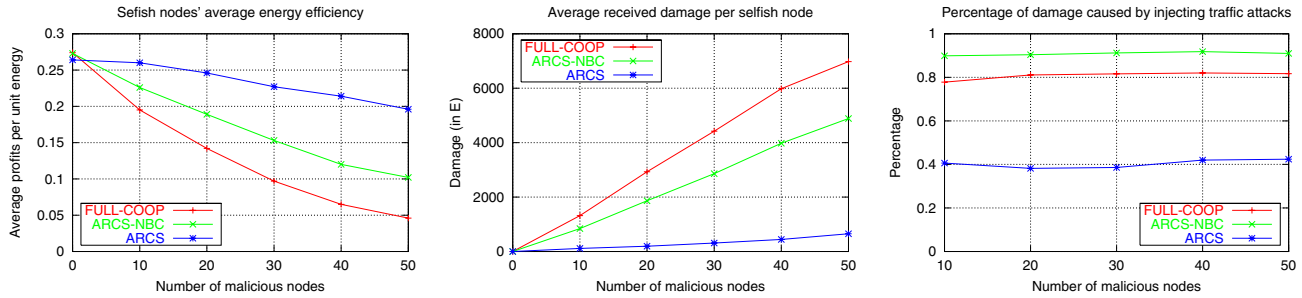
1. When node S initiates a route discovery, it will also append its blacklist in the route request packet. After an intermediate node A has received the request packet, it will update its own record $B_{lacklist}(S, A)$ using the received blacklist.
2. When an intermediate node A receives a route request packet which originates from S and A is not this request's destination, A first checks the following conditions: 1) A has never seen this request before; 2) A is not in S's blacklist; 3) $B(S, A) < B_{th}(S, A)$; 4) no nodes that have been appended to the request packet are in A's blacklist; 5) A has not forwarded any request for S in the last $T_{interval}(S, A)$ interval, where $T_{interval}(S, A)$ is the minimum interval specified by A to indicate that A will forward at most one route request for S in each $T_{interval}(S, A)$ interval. A will broadcast the request if and only if all of the above conditions can be satisfied, otherwise, A will discard the request.
3. During a discovered route is being returned to the requester S, each intermediate node A on the route appends the following information to the returned route: the subset of its blacklist that is not known by S, the value of $B_{th}(S, A)$ if not known by S, and the value of $D_{ebit}(S, A)$. After S has received the route, for each node A on the discovered route, it updates the corresponding blacklist $B_{lacklist}(A, S)$ and the value of $B_{th}(S, A)$.

IV. Simulation Studies

In the simulation, nodes are randomly deployed inside a rectangular area of 1000m $\times$ 1000m. Each node moves randomly according to the *random waypoint* model [13] with maximum speed $v_{max} = 20m/s$. The physical layer assumes a fixed transmission range model, where two nodes can directly communicate with each other successfully only if they are in each other's transmission range. The MAC layer protocol simulates the IEEE 802.11 DCF with a four-way handshaking mechanism [14]. The transmission range is fixed to be 250m.

Each selfish node randomly picks another selfish node as the receiver and packets are scheduled to be generated according to a Poisson process. Similarly, each attacker also randomly picks another attacker as the receiver to send packets. The total number of good nodes is fixed to be 100, and the total number of attackers varies from 0 to 50. Among those attackers, 1/3 launch dropping packets attacks which drop all packets passing through them whose sources are not malicious, 1/3 launch emulating link breakage attacks which emulate link breakage once receiving packet forwarding request from selfish nodes, and 1/3 launch injecting traffic attacks. For each selfish node or attacker that does not launch injecting traffic attacks, the average packet inter-arrival time is 2 seconds, while for attackers launching injecting traffic attacks, the average packet inter-arrival time is 0.1 second. In the simulations, all data packets have the same size.

Based on selfish nodes' forwarding decision, three types of systems have been implemented in the simulations: the proposed proposed system called "ARCS" (attack-resilient cooperation stimulation); the proposed system without balance constraint (i.e., cooperation degree is set to be infinity) called "ARCS-NBC"; and a fully-cooperative system called "FULL-COOP". In "ARCS", all selfish nodes behave in the way as described in Section III, where $B_{threshold}$ is set to be $50E$ and $T_{interval}$ is set to be 100 seconds. In "ARCS-NBC", the same strategies as in "ARCS" have been used to detect launching dropping packets attacks and emulating link breakage attacks, but now (5) is not a necessary condition to forward packets for other nodes, and a selfish node will unconditionally forward packets for those nodes which have not been marked as malicious by it. In "FULL-COOP", all selfish nodes will unconditionally forward packets for other nodes, and no attackers detection and punishment mechanisms have been used. In this paper, the following metrics are used to



(a) Energy efficiency comparison (b) Average damage comparison (c) Effects of injecting traffic attacks

Fig. 2. Performance comparison among the three systems

evaluate system performance: the energy efficiency of selfish nodes and the average damage received per selfish node.

In our simulations, each configuration has been run 10 independent rounds using different random seeds, and the result are averaged over all the rounds. In the simulations, we set $\alpha_S = 1$, $\beta_S = 0.5$, and $T_{interval}$ to be 100 seconds for any selfish node S. The running time for each round is 5000 seconds. Fig. 2 shows the performance comparison among the three systems: ARCS, ARCS-NBC, and FULL-COOP, where in ARCS, B_{th} and LB_{th} are both set to be $60E$. From the selfish nodes' energy efficiency comparisons (Fig. 2(a)) we can see that ARCS has much higher efficiency than ARCS-NBC and FULL-COOP when there exist attackers. The average damage comparison (Fig. 2(b)) shows that in ARCS the damage that can be caused by attackers is much lower than in other two systems, and increases very slowly with the increase of attacker number.

From the results shown in Fig. 2(a) and Fig. 2(b) we can also see that although ARCS-NBC has a lot of improvement over FULL-COOP by introducing mechanisms to detect dropping packet and emulating link breakage attacks, its performance is still much worse than ARCS. The reason is that ARCS-NBC cannot detect and punish those attackers which launch injecting traffic attacks, so a large portion of energy has been wasted to forward packets for those nodes. Fig. 2 (c) illustrates the different effects of injecting traffic attacks in the three systems, where the y-axis shows the percentage of damage caused by injecting traffic attacks to the network. From these results we can see that in ARCS, only about 40% percentage of damage is caused by injecting traffic attacks, in FULL-COOP this percentage increases to around 80%, while in ARCS-NBC the percentage increases to more than 90%, although the overall damage caused by all attackers to the selfish nodes in ARCS-NBC is less than that in FULL-COOP. These results explain why ARCS-NBC has much worse performance than ARCS, and clearly show that how necessary it is to introduce mechanisms to defend against such injecting traffic attacks.

V. Conclusion

In this paper we investigate the issues of cooperation stimulation and security in self-organized ad hoc networks, and proposed an anti-attack cooperation stimulation system to enforce cooperation among selfish nodes and to defend against various attacks launched by attackers. Both analysis and simulation results have shown that in the proposed system, the damage that can be caused by attackers is bounded, and the cooperation among selfish nodes can be enforced through introducing a positive cooperation degree. Another key property of the proposed system is that it is fully distributive, completely self-organizing, and does not require any tamper-proof hardware or central management points.

References

[1] J.-P. Hubaux and T. Gross and J.-Y. Le Boudec and M. Vetterli, "Toward Self-Organized Mobile Ad Hoc Networks: The Terminodes Project," *IEEE Communications Magazine*, Jan. 2001.

[2] L. Buttyan and J. P. Hubaux, "Enforcing Service availability in mobile Ad-hoc Network," in *MobiHOC*, August 2000.

[3] L. Buttyan and J.-P. Hubaux, "Stimulating Cooperation in Self-organizing Mobile Ad Hoc Networks," *Mobile Networks and Applications*, vol. 8, no. 5, pp. 579 – 592, Oct. 2003.

[4] S. Zhong, J. Chen, and Y. R. Yang, "Sprite: A Simple, Cheat-Proof, Credit-Based System for Mobile Ad-Hoc Networks," in *INFOCOM 2003*, 2003.

[5] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," in *Mobicom 2000*, August 2000, pp. 255–265.

[6] P. Michiardi and R. Molva, "Core: a COllaborative REputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," in *IFIP - Communications and Multimedia Security Conference*, 2002.

[7] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol," in *Mobihoc*, 2002, pp. 226 – 236.

[8] L. Zhou and Z. Haas, "Securing Ad Hoc Networks," *IEEE Network Magazine*, vol. 13, no. 6, pp. Nov./Dec., 1999.

[9] J. P. Hubaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," in *MobiHOC 2001*, May 2001.

[10] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," in *MobiCom 2002*, Atlanta, GA, USA, Sep. 2002.

[11] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," in *WiSe*, San Diego, CA, USA, Sep. 2003.

[12] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks," in *IEEE Infocom*, 2003.

[13] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing," In *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

[14] IEEE Computer Society LAN MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1007," The Institue of Electrical and Electrics Engineers.