Game Theoretic Analysis of Cooperation Stimulation and Security in Autonomous Mobile Ad Hoc Networks

Wei Yu and K.J. Ray Liu, *Fellow, IEEE*

Abstract—In autonomous mobile ad hoc networks, nodes belong to different authorities and pursue different goals; therefore, cooperation among them cannot be taken for granted. Meanwhile, some nodes may be malicious, whose objective is to damage the network. In this paper, we present a joint analysis of cooperation stimulation and security in autonomous mobile ad hoc networks under a game theoretic framework. We first investigate a simple yet illuminating two-player packet forwarding game and derive the optimal and cheat-proof packet forwarding strategies. We then investigate the secure routing and packet forwarding game for autonomous ad hoc networks in noisy and hostile environments and derive a set of reputation-based cheat-proof and attack-resistant cooperation stimulation strategies. When analyzing the cooperation strategies, besides Nash equilibrium, other optimality criteria, such as Pareto optimality, subgame perfection, fairness, and cheat-proofing, have also been considered. Both analysis and simulation studies have shown that the proposed strategies can effectively stimulate cooperation among selfish nodes in autonomous mobile ad hoc networks under noise and attacks, and the damage that can be caused by attackers is bounded and limited.

Index Terms—Autonomous mobile ad hoc networks, security, cooperation stimulation, game theory.

1 Introduction

A *mobile ad hoc network* is a group of mobile nodes that do not require a fixed network infrastructure in which nodes can communicate with each other out of their direct transmission ranges through cooperatively forwarding packets. In traditional emergency or military applications, nodes in a mobile ad hoc network usually work in a fully cooperative way. Recently, emerging applications of mobile ad hoc networks have also been envisioned in civilian usage [1], where nodes typically do not belong to the same authority and may not pursue common goals. Consequently, fully cooperative behaviors, such as unconditionally forwarding packets for each other, cannot be taken for granted. On the contrary, in order to save limited resources, nodes tend to be "selfish." We refer to such networks as *autonomous mobile ad hoc networks*.

Before ad hoc networks can be successfully deployed in an autonomous way, however, the issue of *node cooperation* must be resolved first. In the literature, many schemes have been proposed to stimulate node cooperation in ad hoc networks [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16]. One way to stimulate cooperation among selfish nodes is to use payment-based methods, such as those proposed in [2], [3], [4], [5], [6]. Although these schemes can effectively stimulate cooperation among selfish nodes, the requirement of tamper-proof hardware or central billing services greatly limits their potential applications.

Another way to stimulate cooperation among selfish nodes is to use reputation-based methods with necessary monitoring [7], [8], [9], [10]. In [7], a reputation-based system was proposed to mitigate nodes' misbehavior, where each node launches a "watchdog" to monitor its neighbors' packet forwarding activities. Following [7], Core was proposed to enforce cooperation among selfish nodes [8] and CONFIDANT was proposed to detect and isolate misbehaving nodes and, thus, make it unattractive to deny cooperation [9]. Recently, ARCS was proposed to simultaneously stimulate cooperation among selfish nodes and defend against various attacks [10]. Meanwhile, efforts have also been made toward mathematically analyzing cooperation in autonomous ad hoc networks in a game theoretic framework, such as [11], [12], [13], [14], [15], [16].

In this paper, we also focus on designing reputation-based cooperation stimulation strategies for autonomous mobile ad hoc networks under a game theoretic framework. However, there are several major differences to distinguish our work from the existing work, such as [11], [12], [13], [14], [15], [16]. First, instead of focusing only on selfish behavior, in our analysis, the effect of possible malicious behavior has also been incorporated. Second, we have addressed these issues under more realistic scenarios by considering error-prone communication channels, while most existing cooperation schemes can only work well in ideal environments. Third, we have also fully explored possible cheating behavior and derived cheat-proof strategies, while most existing work require nodes to honestly report their private information. Fourth, in our analysis, besides Nash equilibrium, other optimality criteria, such as Pareto optimality, cheat-proofing, and fairness, have also been applied.

We first studied a simple yet illuminating two-player packet forwarding game and investigated the Nash equilibria. Since this game usually has multiple equilibria, we

- *The authors are with the Department of Electrical and Computer Engineering and The Institute for Systems Research, University of Maryland, College Park, MD 20742.*
E-mail: weiyu@glue.umd.edu, kjrliu@isr.umd.edu.

Manuscript received 26 June 2006; revised 27 Jan. 2006; accepted 5 Sept. 2006; published online 7 Feb. 2007.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-0187-0605.
Digital Object Identifier no. 10.1109/TMC.2007.1026.

then investigated how to apply extra optimality criteria, such as subgame perfection, Pareto optimality, fairness, and cheat-proofing, to further refine the obtained Nash equilibrium solutions. Finally, a unique Nash equilibrium solution is derived which suggests that a node should not help its opponent more than its opponent has helped it. The analysis is then extended to handle multinode scenarios in noisy and hostile environments by modeling the dynamic interactions between nodes as secure routing and packet forwarding games. By taking into consideration the difference between the two-node case and multinode case, we have also derived attack-resistant and cheat-proof cooperation stimulation strategies for autonomous mobile ad hoc networks. Both analysis and simulation studies have shown that the proposed strategies can effectively stimulate cooperation under noise and attacks, and the damage that can be caused by attackers is bounded and limited.

The rest of this paper is organized as follows: Section 2 studies the two-player packet forwarding game. Section 3 investigates the secure routing and packet forwarding game for autonomous ad hoc networks in noisy and hostile environments. In Section 4, extensive simulations have been conducted to evaluate the performance of the proposed strategies under various scenarios. Finally, Section 5 concludes this paper.

2 Optimal Strategies in a Two-Player Packet Forwarding Game

2.1 Game Model

In this paper, we focus on the most basic networking function, namely, packet forwarding. We first study a simple yet illuminating two-player packet forwarding game. In this game, there are two players, denoted by $N = \{1, 2\}$. Each player needs its opponent to forward a certain number of packets in each stage. For each player i, the cost to forward a packet is c_i, and the gain it can get for each packet that its opponent has forwarded for it is g_i. To simplify our illustration, we assume that all packets have the same size. Here, the cost can be the consumed energy and the gain is usually application-specific. It is reasonable to assume that $g_i \geq c_i$ and there exists a c_{max} with $c_i \leq c_{max}$. Let B_i be the number of packets that player i will request its opponent to forward in each stage. Here, B_i, c_i, and g_i are player i's private information, which is not known to its opponents unless player i reports them (either honestly or dishonestly).

In each stage, let $A_1 = \{0, 1, \dots, B_2\}$ denote the set of actions that player 1 can take and let $A_2 = \{0, 1, \dots, B_1\}$ denote the set of actions that player 2 can take. That is, $a_i \in A_i$ denotes that player i will forward a_i packets for its opponent in this stage. We refer to an action profile $a = (a_1, a_2)$ as an *outcome* and denote the set $A_1 \times A_2$ of outcomes by A. Then, in each stage, players' payoffs are calculated as follows, provided the action profile a being taken:

$$u_1(a) = a_2 \times g_1 - a_1 \times c_1, \quad (1)$$

$$u_2(a) = a_1 \times g_2 - a_2 \times c_2. \quad (2)$$

That is, the payoff of a player is the difference between the total gain it obtained with the help of its opponent and the total cost it spent to help its opponent. We refer to $u(a) = (u_1(a), u_2(a))$ as the payoff profile associated with the action profile a.

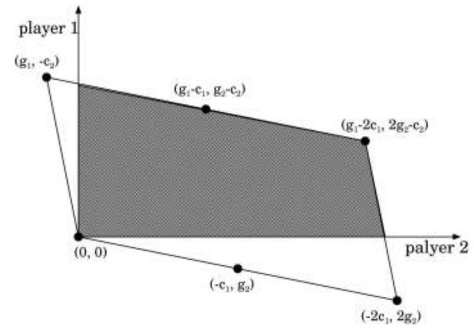


Fig. 1. Feasible and enforceable payoff profiles.

It is easy to check that, if this game will only be played for one time, the only Nash equilibrium (NE) is $a^* = (0, 0)$. According to the backward induction principle [17], this is also true when the stage game will be played for finite times with game termination time known to both players. Therefore, in such scenarios, for each player, its only optimal strategy is to always play noncooperatively.

However, in most situations, these two players may interact many times and no one can know exactly when its opponent will quit the game. Next, we show that, under a more realistic setting, besides the noncooperative strategy, cooperative strategies can also be obtained. Let G denote the repeated version of the above one-stage packet forwarding game. Let s_i denote player i's behavior strategy, and let $s = (s_1, s_2)$ denote the strategy profile. Next, we consider the following two utility functions:

$$U_i(s) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T} u_i(s), \quad U_i(s, \delta) = (1 - \delta) \sum_{t=0}^{\infty} \delta^t u_i(s). \quad (3)$$

Utility function $U_i(s)$ can be used when the game will be played for infinite times and the discounted version $U_i(s, \delta)$ can be used when the game will be played for finite times, but no one knows the exact termination time. Here, the discount factor δ (with $0 < \delta < 1$) characterizes each player's expected playing time. Since, in general, the results obtained based on $U_i(s)$ can also be applied to the scenarios when $U_i(s, \delta)$ is used as long as δ approaches 1, in this section, we will mainly focus on $U_i(s)$.

Now, we analyze possible NE for the game G with utility function $U_i(s)$. According to the Folk theorem [17], for every feasible and enforceable payoff profile, there exists at least one NE to achieve it, where the set of feasible payoff profiles for the above game is

$$V_0 = \text{convex hull}\{v \mid \exists a \in A \text{ with } u(a) = v\} \quad (4)$$

and the set of enforceable payoff profiles, denoted by V_1, is

$$\begin{aligned} V_1 = &\{v \mid v \in V_0 \text{ and } \forall i : v_i \geq \underline{v}_i, \\ &\text{where } \underline{v}_i = \min_{a_{-i} \in A_{-i}} \max_{a_i \in A_i} u_i(a_{-i}, a_i)\}. \end{aligned} \quad (5)$$

Fig. 1 depicts these sets for the game with $B_1 = 1$ and $B_2 = 2$, where the vertical axis denotes player 1's payoff and the horizontal axis denotes player 2's payoff. The payoff profiles inside the convex hull of

$$\{(0, 0), (g_1, -c_2), (g_1 - 2c_1, 2g_2 - c_2), (-2c_1, 2g_2)\}$$

(including the boundaries) are the set of feasible payoff profiles V_0; the set of payoff profiles inside the shading area (including the boundaries) are the set of feasible and enforceable payoff profiles V_1. We can easily check that, as long as $g_1 g_2 > c_1 c_2$, there exists an infinite number of NE. To simplify our illustration, in this paper, we will use $x = (x_1, x_2)$ to denote the set of NE strategies corresponding to the enforceable payoff profile $(x_2 g_1 - x_1 c_1, x_1 g_2 - x_2 c_2)$.

2.2 Nash Equilibrium Refinements

Based on the above analysis, we can see that the infinitely repeated game G may have an infinite number of NE. It is also easy to check that not all the obtained NE payoff profiles are simultaneously acceptable to both players. For example, the payoff profile $(0, 0)$ will not be acceptable from both players' point of view if they are rational. Next, we show how to perform *equilibrium refinement*, that is, how to introduce new optimality criteria to eliminate those NE solutions which are less rational, less robust, or less likely. Specifically, the following optimality criteria will be considered: *Pareto optimality*, *subgame perfection*, *proportional fairness*, and *absolute fairness*.

2.2.1 Subgame Perfection

Our first step toward refining the NE solutions is to rule out those empty threats based on more credible punishments, known as *subgame perfect equilibrium*. According to the perfect Folk theorem [17], every strictly enforceable payoff profile $v \in V_2$ is a subgame perfect equilibrium payoff profile of the game G, where

$$\begin{aligned} V_2 = &\{v \mid v \in V_0 \text{ and } \forall i : v_i > \underline{v}_i, \\ &\text{where } \underline{v}_i = \min_{a_{-i} \in A_{-i}} \max_{a_i \in A_i} u_i(a_{-i}, a_i)\}. \end{aligned} \tag{6}$$

That is, after applying the criterion of subgame perfection, only a small set of NE are removed.

2.2.2 Pareto Optimality

Our second step toward refining the set of NE solutions is to apply the criterion of *Pareto optimality*.[1] It is easy to check that only those payoff profiles lying on the boundary of the set V_0 could be Pareto optimal. Let V_3 denote the subset of feasible payoff profiles which are also Pareto optimal. For the case depicted in Fig. 1, V_3 is the set of payoff profiles which lie on the segment between $(g_1, -c_2)$ and $(g_1 - 2c_1, 2g_2 - c_2)$ and on the segment between $(g_1 - 2c_1, 2g_2 - c_2)$ and $(-2c_1, 2g_2)$. After applying the criterion of Pareto optimality, although a large portion of NE have been removed from the feasible set, there still exist an infinite number of NE. Let $V_4 = V_3 \cap V_2$.

2.2.3 Proportional Fairness

Next, we try to further refine the solution set based on the criterion of proportional fairness. Here, a payoff profile is proportionally fair if $U_1(s)U_2(s)$ can be maximized, which can be achieved by maximizing $u_1(s)u_2(s)$ in each stage. Then, we can reduce the solution set V_4 to a unique point as follows:

$$x^* = \begin{cases} \left(\frac{\frac{c_2}{g_2}+\frac{g_1}{c_1}}{2} B_1, B_1\right) & \text{if} \quad \frac{B_1}{B_2} < \frac{2}{\frac{c_2}{g_2}+\frac{g_1}{c_1}} \\ (B_2, B_1) & \text{if} \quad \frac{2}{\frac{c_2}{g_2}+\frac{g_1}{c_1}} \le \frac{B_1}{B_2} \le \frac{\frac{c_1}{g_1}+\frac{g_2}{c_2}}{2} \\ (B_2, \frac{\frac{c_1}{g_1}+\frac{g_2}{c_2}}{2} B_2) & \text{if} \quad \frac{B_1}{B_2} > \frac{\frac{c_2}{g_2}+\frac{g_1}{c_1}}{2}. \end{cases} \tag{7}$$

2.2.4 Absolute Fairness

In many situations, absolute fairness is also an important criterion. We first consider *absolute fairness in payoff*, which refers to the fact that the payoff of these two players should be equal. Then, we can reduce the solution set V_4 to a unique point as follows:

$$x^* = \begin{cases} \left(\frac{g_1+c_2}{g_2+c_1} B_1, B_1\right) & \text{if} \quad \frac{B_1}{B_2} \le \frac{g_2+c_1}{g_1+c_2}, \\ \left(B_2, \frac{g_2+c_1}{g_1+c_2} B_2\right) & \text{if} \quad \frac{B_1}{B_2} \ge \frac{g_2+c_1}{g_1+c_2}. \end{cases} \tag{8}$$

Another similar criterion is *absolute fairness in cost*, which refers to the fact that the cost incurred to both players should be equal. Then, we can also reduce the solution set V_4 to a unique point as follows:

$$x^* = \begin{cases} \left(\frac{c_2}{c_1} B_1, B_1\right) & \text{if} \quad \frac{B_1}{B_2} \le \frac{c_1}{c_2}, \\ \left(B_2, \frac{c_1}{c_2} B_2\right) & \text{if} \quad \frac{B_1}{B_2} \ge \frac{c_1}{c_2}. \end{cases} \tag{9}$$

2.3 Optimal and Cheat-Proof Packet Forwarding Strategies

In Section 2.2, we have obtained several unique solutions after applying different optimality criteria to refine the original solutions. However, all these solutions involve some private information reported by each selfish player. Due to players' selfishness, honestly reporting private information cannot be taken for granted and players may tend to cheat whenever they believe cheating can increase their payoffs. In this paper, we refer to a NE as *cheat-proof* if no player can further increase its payoff by revealing false private information to its opponents. In the Appendix, we have examined the three solutions (7), (8), and (9). The analysis shows that none of them is cheat-proof with respect to the private information c_i and g_i. Since all these unique solutions are strongly Pareto optimal, the increase of its opponent's payoff will lead to the decrease of its own payoff. Therefore, players have no incentive to honestly report their private information. On the contrary, they will cheat whenever cheating can increase their payoff.

What is the consequence if both players will cheat with respect to c_i and g_i? Let us first examine the solution (7). In this case, based on the analysis in the Appendix, both players will report a c_i/g_i value as high as possible. Since we have assumed $g_i \ge c_i$ and $c_i \le c_{max}$, both players will set $g_i = c_i = c_{max}$ and the solution (7) will become the following form:

$$x^* = (\min(B_1, B_2), \min(B_1, B_2)). \tag{10}$$

After applying similar analysis on the solutions (8) and (9), we can also see that they will also converge to the form (10). Accordingly, the corresponding payoff profile is

$$v^* = ((g_1 - c_1)\min\{B_1, B_2\}, (g_2 - c_2)\min\{B_1, B_2\}). \tag{11}$$

It is easy to check that solution (10) forms Nash equilibrium, is Pareto optimal, and is cheat-proof with respect to private information c_i and g_i.

[1] Given a payoff profile $v \in V_0$, v is said to be Pareto optimal if there is no $v' \in V_0$ for which $v'_i > v_i$ for all $i \in N$; v is said to be strongly Pareto optimal if there is no $v' \in V_0$ for which $v'_i \ge v_i$ for all $i \in N$ and $v'_i > v_i$ for some $i \in N$ [17].

From the above analysis, we have learned that nodes can report false c_i and g_i to increase their payoff. Recall that B_i is also regarded as private information reported by player i. Then, a natural question is: *In the obtained solutions (7), (8), (9), and (10), can player i further increase its payoff by reporting false B_i value?* The answer is "NO" after examining all these four unique solutions. In other words, all these unique solutions are cheat-proof with respect to the private information B_i.

Next, we analyze why these solutions are cheat-proof with respect to B_i. Let us first determine whether player 1 can further increase its payoff by reporting a high false B_1value. Here, we use B_1 to denote the actual number of packets that player 1 needs its opponent to forward in each stage; that is, player 1 cannot get any extra gain if player 2 will forward more packets than B_1. Let B'_1 be a certain value that player 1 may report to its opponent. Now, let us examine the consequence when player 1 reports a false $B'_1 > B_1$. By checking the four solutions, we can see that such a B'_1 can never decrease x_1 (i.e., the number of packets that player 1 needs to forward for player 2). Further, we can see that, when x_1 is increased upon reporting a false $B'_1 > B_1$, the optimal x_2 is always B_1. Therefore, reporting a $B'_1 > B_1$ can never increase player 1's payoff. How about reporting a false $B'_1 < B_1$? After checking all these solutions, we can see that such actions will either cause no effect on these solutions or will decrease x_2. Although such actions may also decrease x_1, the penalty due to the decrease of x_2 is always less than the gain due to the decrease of x_1. Therefore, reporting a $B'_1 < B_1$ can never increase player 1's payoff. Due to the symmetry between these two players, it is easy to check that reporting a false B'_2 can never increase player 2's payoff too.

Based on the above analysis, we can conclude that, in the two-player packet forwarding game, in order to maximize its own payoff and be resistant to possible cheating behavior, a player should not forward more packets than its opponent does for it. Specifically, *for each player $i \in N$, in each stage, it should forward $\min(B_1, B_2)$ packets for its opponent unless there was a previous stage in which its opponent forwarded less than $\min(B_1, B_2)$ packets for it, in which case it will stop forwarding packets for its opponent forever*. We refer to the above strategy as **two-player cheat-proof packet forwarding strategy**.

2.4 Discussion

The strategies proposed in [11], [12] may look similar to the one described above. In [11], Srinivasan et al. studied the cooperation in ad hoc networks by focusing on the energy-efficient aspects of cooperation, where, in their solution, the nodes are classified into different energy classes and the behavior of each node depends on the energy classes of the participants of each connection. They have demonstrated that, if two nodes belong to the same class, they should apply the same packet forwarding ratio. However, they require nodes to *honestly* report their classes, and a node can easily cheat to increase its own performance, such as the approach mentioned in [16] (Section 8). Meanwhile, using normalized throughput alone as the performance metric may not be a good choice in general, as to be explained in the Section 3.5.

In [12], Urpi et al. claimed that it is not possible to force a node to forward more packets than it sends on average (Lemma 6.2), and then concluded that cooperation can be enforced in a mobile ad hoc network, provided that enough members of the network agree on it, and if no node has to forward more traffic that it generates (Theorem 6.3). However, the above analysis has shown that a strategy profile can still be enforceable even this may require a node to forward more packets than it sends on average, as illustrated in solutions (7), (8), and (9). Second, in mobile ad hoc networks, due to the multihop nature, in general, the number of packets a node forwards should be much more than the number of packets it generates. Accordingly, their strategy is not applicable in most scenarios.

The works presented in [5], [6] are also related to ours in the sense that cheating behavior has also been considered. They have proposed auction-based schemes to stimulate packet forwarding participation, where, by using VCG-based second price auctioning, these schemes force selfish nodes to honestly report their true private information (such as cost) to maximize their profit. However, in their schemes, a trusted third-party auctioneer is required per route selection and central banking services are needed to handle billing information, which usually cannot be satisfied in a mobile ad hoc network. In our work, we focus on the scenario that neither a trusted third-party auctioneer nor central banking service is available.

3 Secure Routing and Packet Forwarding Game

3.1 System Description and Game Model

Next, we will investigate how to stimulate cooperation for autonomous mobile ad hoc networks in noisy and hostile environments. We focus on the scenario that the network will keep alive for a relatively long time, and there exist a finite number of users, such as students in a campus. Each user will stay in the network for a reasonably long time, but is allowed to leave and reconnect to the network when necessary. We assume that each legitimate user has a unique registered and verifiable identity (e.g., a public/private key pair), which is issued by some central authority and will be used to perform necessary access control and authentication. We focus on an information-push model where it is the source's duty to guarantee the successful delivery of packets to their destinations. For each user, forwarding a packet will incur some cost, and a successful delivery of a packet originating from it to its destination can bring some gain. Here, the cost can be the consumed energy and the gain is usually user and/or application-specific.

Since ad hoc networks are usually deployed in adversarial environments, some nodes may be malicious, whose goal is to cause damage to other nodes. In this paper, we focus on insider attackers, that is, the attackers also have legitimate identities. Preventing outside attackers from entering the network can be easily achieved through employing necessary access control and communicating through shared secret channels. Instead of forcing all users to act fully cooperatively (which has been shown not achievable in some situations [16], [6]), our goal is to stimulate cooperation among selfish nodes as much as possible.

In general, not all packet forwarding decisions can be perfectly executed. For example, when a node has decided to help another node to forward a packet, the packet may still be dropped due to link breakage or the transmission may fail due to channel errors.[2] In this paper, we use p_e to denote the packet dropping probability owing to noise. We also assume that some underlying monitoring schemes, such as those presented in [18], [10], have been launched, where the source can know whether its packets have been successfully delivered through end-to-end acknowledgement and can detect who has dropped its packets.

In order to formally analyze cooperation and security in such networks, we model the interactions among nodes as a **secure routing and packet forwarding game**:

- **Players**: The finite number of users in the network, denoted by N.
- **Types**: Each player $i \in N$ has a type $\theta_i \in \Theta$, where $\Theta = \{\mathit{selfish}, \mathit{malicious}\}$. Let N_s denote the set of selfish players and $N_m = N - N_s$ denote the set of (insider) attackers.
- **Strategy space**: Routing and packet forwarding are jointly considered, and each packet forwarding transaction is divided to three stages:

    1. **Route participation stage**: For each player, after receiving a request asking it to be on a certain route, it can either *accept* or *refuse* this request.
    2. **Route selection stage**: For each player who has a packet to send, after discovering a valid route, it can either *use* or *not use* this route to send the packet.
    3. **Packet forwarding stage**: For each relay, once it has received a packet requesting it to forward, its decision can be either *forward* or *drop* this packet.

- **Cost**: For any player $i \in N$, transmitting a packet, either for itself or for the others, incurs cost c_i.
- **Gain**: For each selfish player $i \in N_s$, if a packet originated from it can be successfully delivered to its destination, it can get gain g_i, where $g_i \geq c_i$.
- **Utility**: For each player $i \in N$, $T_i(t)$ denotes the number of packets that player i needs to send by time t, $S_i(t)$ denotes the number of packets that have been successfully delivered to their destinations by time t with player i being the source, $F_i(j,t)$ denotes the number of packets that i has forwarded for player j by time t, and $F_i(t) = \sum_{j \in N} F_i(j,t)$. Let $W_i(j,t)$ denote the total number of useless packet transmissions that player i has caused to player j by time t due to i dropping the packets forwarded by j. Let t_f be the lifetime of this network. Then, we model the players' utility as follows:

    1. For any selfish player $i \in N_s$, its objective is to maximize $U_i^s(t_f)$ with

    $$U_i^s(t_f) = \frac{S_i(t_f) g_i - F_i(t_f) c_i}{T_i(t_f)}. \tag{12}$$

    2. For any malicious player $j \in N_m$, its objective is to maximize $U_j^m(t_f)$ with

    $$U_j^m(t_f) = \frac{1}{t_f} \sum_{i \in N_s} \big(W_j(i,t_f) + F_i(j,t_f) \big) c_i - F_j(t_f) c_j. \tag{13}$$

    If the game will be played for an infinite duration, then their utilities will become $\lim_{t_f \to \infty} U_i^s(t_f)$ and $\lim_{t_f \to \infty} U_j^m(t_f)$, respectively.

On the right-hand side of (12), the numerator denotes the net profit (i.e., total gain minus total cost) that the selfish node i obtained, and the denominator denotes the total number of packets that i needs to send. This utility function represents the average net profit that i can obtain per packet. We can see that maximizing (12) is equivalent to maximizing the total number of successful deliveries subject to the total available cost constraint. If $c_i = 0$, this is equal to maximizing the throughput. The numerator of the right-hand side of (13) represents the net damage caused to the other nodes by j. Since, in general, this value may increase monotonically, we normalize it using the network lifetime t_f. Now, this utility function represents the average net damage that j causes to the other nodes per time unit. From (13), we can see that, in this game, the attackers' goal is to waste the selfish nodes' cost (or energy) as much as possible. The attackers are allowed to collude to maximize their performance. Other possible alternatives, such as minimizing the others' payoff, will be discussed in Section 3.4.

3.2 Attack-Resistant and Cheat-Proof Cooperation Stimulation Strategies

Based on the system description in Section 3.1, we can see that the multinode scenario is much more complicated than the two-node scenario, and directly applying the two-player cooperation strategies to multinode scenarios may not work. In this section, we first explore the challenges to stimulate cooperation for autonomous mobile ad hoc networks in noisy and hostile environments, then devise attack-resistant and cheat-proof cooperation stimulation strategies.

First, in autonomous mobile ad hoc networks, the repeated game model is not applicable any more. For example, a source may request different nodes to forward packets at different times and may act as a relay for different sources. Meanwhile, the request rates of each node to other nodes are usually variable, which can be caused either by its inherent variable traffic generation rate or by mobility. A direct consequence of such a nonrepeated model is that favors cannot be simultaneously granted. In [19], Dawkins demonstrated that reciprocal altruism is beneficial for every ecological system when favors are granted simultaneously. However, when favors cannot be granted simultaneously, altruism may not guarantee satisfactory future payback, especially when the future is unpredictable. This makes cooperation stimulation in autonomous mobile ad hoc networks an extremely challenging task.

Second, in wireless networks, noise is inevitable and can cause severe trouble. For a two-player cheat-proof packet forwarding strategy, if some packets are dropped due to noise, the game will be terminated immediately and the performance will be degraded drastically. This will also happen in most existing cooperation enforcement schemes, such as [11], [16]. In these schemes, noise can easily cause the collapse of the whole network, where, finally, all nodes will act noncooperatively. Distinguishing the misbehavior

[2] We refer to all those factors which can cause imperfect decision execution as *noise*, including those uncertainty factors such as channel errors, link breakages, congestion, etc.

caused by noise from that caused by malicious intention is a challenging task.

Third, since autonomous mobile ad hoc networks are usually deployed in adversarial environments, some nodes may even be malicious. If there exist only selfish nodes, stimulating cooperation will be much easier according to the following logic as demonstrated in [16]: Misbehavior can result in the decrease of service quality experienced by some other nodes, which may consequently decrease the quality of service provided by them; this quality degradation will then be propagated back to the misbehaving nodes. Therefore, selfish nodes have no incentive to intentionally behave maliciously in order to enjoy a high quality of service. However, this is not the case when some nodes are malicious. Since the attackers' goal is to degrade the network performance, such quality degradation is exactly what they want to see. This makes cooperation stimulation in hostile environments extremely challenging. Unfortunately, malicious behaviors have been heavily overlooked when designing cooperation stimulation strategies.

In the rest of this section, we study how to combat these challenges. We partition the secure routing and packet forwarding game into a set of subgames, where each subgame is played by a pair of nodes. To effectively stimulate cooperation among selfish nodes in noisy and hostile environments, a credit mechanism is introduced to alleviate the effect of nonsimultaneous favor return, and a statistical attacker detection mechanism is introduced to distinguish malicious behavior from misbehavior caused by noise.

We first introduce the credit mechanism. For any two nodes $i, j \in N$, we define $D_i(j,t)$ as follows:

$$D_i(j,t) = F_i(j,t) - F_j(i,t). \tag{14}$$

Now, consider the following strategy: Each node i will limit the number of packets that it will forward for any other node j in such a way that the total number of packets that i has forwarded for j by any time t should be no more than $F_j(i,t) + D_i^{max}(j,t)$, that is,

$$D_i(j,t) \le D_i^{max}(j,t). \tag{15}$$

Here, $D_i^{max}(j,t)$ is a threshold set by i for two purposes: 1) stimulate cooperation between i and j, and 2) limit the possible damage that j can cause to i. By letting $D_i^{max}(j,t)$ be positive, i agrees to forward some extra packets for j without getting instant payback. Meanwhile, unlike acting fully cooperatively, the extra number of packets that i will forward for j will be bounded to limit the possible damage when j plays noncooperatively. This is analogous to a credit card system where $D_i^{max}(j,t)$ can be regarded as the credit line that i sets for j at time t. Like a credit card company adjusts credit lines, $D_i^{max}(j,t)$ can also be adjusted by i over time. We refer to $D_i^{max}(j,t)$ as the *credit line* set by i to j.

It is easy to see that an optimal setting of credit lines is crucial to effectively stimulating cooperation in noisy and hostile environments. Next, we illustrate how to set good credit lines. If the request rates between i and j are constant, then setting the credit line to be 1 will be optimal in the sense that no request will be refused when two nodes have the same request rate. However, due to mobility and nodes' inherent variable traffic generation rates, the request rates between i and j are usually variable. In this case, if the credit lines are set to be too small, some requests will be refused even when the average request rates between them are equal. Let $f_i(j,t)$ denote the number of times that i needs j to help forward packets by time t. If we set the credit lines as follows:

$$\begin{aligned} D_i^{max}(j) &= \max_t\{1, f_i(j,t) - f_j(i,t)\}, \\ D_j^{max}(i) &= \max_t\{1, f_j(i,t) - f_i(j,t)\}, \end{aligned} \tag{16}$$

then, except the first several requests, no other requests will be refused when the average request rates between them are equal. If the average request rates between them are not equal, assuming that $\lim_{t\to\infty}\frac{f_i(j,t)}{f_j(i,t)} > 1$, then no matter how large $D_i^{max}(j)$ is, a certain portion of i's requests will have to be refused. This makes sense since j has no incentive to forward more packets for i. This also suggests that arbitrarily increasing credit lines cannot always increase the number of accepted requests. It is worth pointing out that (16) requires $f_i(j,t)$ and $f_j(i,t)$ to be known by i and j. However, such prior knowledge is usually not available since each node may not know a priori the others' request rates. A simple solution to this is to set the credit lines to be reasonably large positive constants, as in our simulations.

It has been shown in [16] that topology will also play a critical role in enforcing cooperation for fixed ad hoc networks and, in most situations, cooperation cannot be enforced. For example, a node in a bad location may never be able to get help from other nodes due to the fact that no one will need it to forward packets. In this work, we focus on *mobile* ad hoc networks. In such networks, a node in a bad location at a certain time may move to a better location later, or vice versa. This suggests that, when a node receives a packet forwarding request from another node, it should not refuse this request *simply* because the requester cannot help it currently, since the requester may be able to help it later.

Next, we introduce a statistical dropping packet attack detection mechanism to distinguish packet dropping caused by malicious behavior from those caused by noise. To simplify our analysis, we first model packet dropping due to noise as follows: For any player i, when it has decided to forward a packet for any other player j, with probability $1 - p_e$, this packet may be dropped due to noise. That is, packet dropping caused by noise is modeled using a Bernoulli random process. p_e can be either estimated by nodes online or trained offline. Let $R_i(j,t)$ denote the number of packets that i has requested j to forward and j has agreed by time t. Based on the Central Limit Theorem [20], for any $x \in \mathcal{R}^+$, we can have

$$\lim_{R_i(j,t)\to\infty} Prob\left(\frac{F_j(i,t) - R_i(j,t)p_e}{\sqrt{R_i(j,t)p_e(1-p_e)}} \ge -x\right) = \Phi(x), \tag{17}$$

where

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-t^2/2}dt. \tag{18}$$

That is, when $R_i(j,t)$ is large, the sufficient statistics $F_j(i,t) - R_i(j,t)p_e$ can be approximately modeled as a random variable with mean 0 and variance

$$R_i(j,t)p_e(1-p_e).$$

Let $isBad_i(j)$ denote i's belief about j's type, where $isBad_i(j) = 1$ indicates that i believes j is malicious, while $isBad_i(j) = 0$ indicates that i believes j is good. Then, the following hypothesis testing rule can be used by i to judge whether j has maliciously dropped its packets with $1 - \Phi(x)$ being the maximum allowable false positive probability:

$$isBad_i(j) = \begin{cases} 1 & \text{if } F_j(i,t) - R_i(j,t)p_e < -x\sqrt{R_i(j,t)p_e(1-p_e)}, \\ 0 & \text{if } F_j(i,t) - R_i(j,t)p_e \geq -x\sqrt{R_i(j,t)p_e(1-p_e)}. \end{cases} \tag{19}$$

By summarizing the above results, we can arrive at the following cooperation stimulation strategies, which we refer to as **Multinode attack-resistant and cheat-proof cooperation strategy:**

In the secure routing and packet forwarding game, for any selfish node $i \in N_s$, initially, i sets $isBad_i(j) = 0$ for all $j \in N$. Then, in different stages, the following strategy will be used by i:

1. *In the route participation stage, if i has been requested by j to be on a certain route, i will accept this request if j has not been marked as malicious by i and (15) holds; otherwise, i will refuse.*
2. *In the route selection stage, a source i will use a valid route with n hops to send packets only if a) no intermediate nodes on this route have been marked as malicious, b) the expected gain is larger than the expected cost, that is, $(1-p_e)^n g_i > nc_i$, and c) this route has the minimum number of hops among all those valid routes with no nodes being marked as malicious by i.*
3. *In the packet forwarding stage, i will forward a packet for j if i has agreed before and j has not been detected as malicious by i; otherwise, i will drop this packet.*
4. *Let $1 - \Phi(x)$ be the maximum allowable false positive probability from i's point of view, then, as long as $R_i(j,t)$ is large for any node $j \in N$ (e.g., larger than 200), the detection rule (19) will be applied by i after each packet forwarding transaction initiated by it.*

3.3 Strategy Analysis under No Attacks

This section analyzes the optimality of the proposed strategies when no attackers exist. We first consider an infinite lifetime situation with $T_i(t) \to \infty$ as $t \to \infty$. The finite lifetime situation will be discussed later. We assume that credit lines are set in such a way that, for any node i,

$$\lim_{t\to\infty} \frac{D_i^{max}(j,t)}{T_i(t)} = 0, \tag{20}$$

and for any pair of nodes i and j, when $\lim_{t\to\infty} \frac{f_i(j,t)}{f_j(i,t)} \leq 1$, at most a finite number of i's requests will be refused by j due to the fact that (15) does not hold. We also assume that, due to mobility, each pair of nodes in the network can meet infinite times when $t_f \to \infty$.

Lemma 1. *For any selfish node $i \in N$ in the secure routing and packet forwarding game with no attackers, once i has received a route participation request from any other node $j \in N$, if (15) holds and the multinode attack-resistant and cheat-proof cooperation strategy is used by player j, then accepting the request is always an optimal decision from player i's point of view.*

Proof. From player i's point of view, refusing the request may cause it to lack enough balance to request player j to forward packets for it in the future (i.e., $D_j(i,t) > D_j^{max}(i,t)$), while agreeing to forward the packet will not introduce any performance loss due to the assumption (20). Therefore, accepting the request is an optimal decision. □

Lemma 2. *In the secure routing and packet forwarding game where some packet forwarding decisions may not be perfectly executed, from the point of view of any player $j \in N$, if the multinode attack-resistant and cheat-proof cooperation strategy is followed by all the other nodes, in the packet forwarding stage intentionally dropping a packet that it has agreed to forward cannot bring it any gain.*

Proof. When a player $j \in N$ intentionally drops a packet that it has agreed to forward for any other player $i \in N$, it cannot get any gain except saving the cost to transmit this packet. However, since player i follows the multinode attack-resistant and cheat-proof cooperation strategy, that is, it will always try to maintain $\lim_{t\to\infty} \frac{F_i(j,t)}{F_j(i,t)} \geq 1$, by dropping this packet, player j also loses a chance to request player i to forward a packet for it. To get the chance back, player j has to forward another packet for player i. Therefore, intentionally dropping a packet cannot bring any gain to player j. □

Theorem 1. *In the secure routing and packet forwarding game with no attackers, the strategy profile that all players follow the multinode attack-resistant and cheat-proof cooperation strategy forms a subgame perfect equilibrium, is cheat-proof, and achieves absolute fairness in cost if $c_i = c$ for all $i \in N$. If $0 < \lim_{t\to\infty} \frac{T_i(t)}{T_j(t)} < \infty$ for any $i, j \in N$, this strategy profile is also strongly Pareto optimal.*

Proof. We first prove that this strategy profile forms a subgame perfect equilibrium. Since this multiplayer game can be decomposed into many two-player subgames, we only need to consider the two-player subgame played by player i and player j. Suppose that player j does not follow the above strategy; that is, either it will refuse to forward packets for player i when it should, it will intentionally drop packets that it has agreed to forward for player i, it will forward more packets than it should for player i, or it will use nonminimum cost routes to send packets. First, from Lemma 1 and Lemma 2, we know that refusing to forward packets for other players when it should or intentionally dropping packets that it has agreed to forward will not introduce any performance gain. Second, forwarding many more packets (i.e., more than $D_i^{max}(j,t)$) than player j has forwarded for it will not increase its own payoff too according to the assumption of credit line selections. Third, using a nonminimum cost route to send a packet will decrease its expected gain. Based on the above analysis, we can conclude that the above strategy profile (the multiplayer attack-resistant and cheat-proof cooperation strategy) forms a Nash equilibrium. To check that the profile is subgame perfect, note that, in every subgame off the equilibrium path, the strategies are either to play noncooperatively forever if player j has dropped a certain number of packets that it has agreed to forward for player i, which is a Nash equilibrium, or still to play the multiplayer cheat-proof packet forwarding strategy, which is also a Nash equilibrium.

Since no private information of g_i and c_i has been involved, based on the analysis presented in Section 2, we can conclude that the proposed cooperation stimulation strategy is cheat-proof.

Since we have $F_i(j,t) - F_j(i,t) < D_i^{max}(j,t)$ for any player $i, j \in N$ and $\lim_{t\to\infty} \frac{D_i^{max}(j,t)}{T_i(t)} = 0$, and we have assumed that $c_i = c$ for all $i \in N$, it always holds that

$$\lim_{t\to\infty} \frac{\sum_{j\in N, j\neq i} F_i(j,t)}{\sum_{j\in N, j\neq i} F_j(i,t)} = 1. \tag{21}$$

That is, this strategy can achieve absolute fairness in cost.

Now, we show that the strategy profile is strongly Pareto optimal. From payoff function (12), we can see that, to increase its own payoff, a player i can either try to increase $S_i(t)$ or decrease $F_i(t)$. However, according to the above strategy, minimum cost routes have been used; therefore, $F_i(t)$ cannot be further decreased without affecting the others' payoff. In order to increase its payoff, the only way that player i can do is to increase $\lim_{t\to\infty} \frac{S_i(t)}{T_i(t)}$, which means that some other players will have to forward more packets for player i. Since all $T_i(t)$s are in the same order, increasing player i's payoff will definitely decrease the other players' payoff. Therefore, the above strategy profile is strongly Pareto optimal. □

In the proof of Theorem 1, we have assumed that 1) $D_i^{max}(j,t)$ is large enough such that forwarding $D_i^{max}(j,t)$ more packets than player j has forwarded for it will not increase its own payoff, and 2) $D_i^{max}(j,t)$ is also small enough such that $\lim_{t\to\infty} \frac{D_i^{max}(j,t)}{T_i(t)} = 0$. If $D_i^{max}(j,t)$ cannot satisfy the above two requirements, the proposed strategy profile is not necessarily a Nash equilibrium. Finding a $D_i^{max}(j,t)$ to satisfy the first requirement is easy, while to satisfy both requirements may be difficult or may even be impossible when nodes' requests rates and mobility patterns are not known a priori. However, our simulation results show that, in many situations, even a nonoptimal $D_i^{max}(j,t)$ can still effectively stimulate cooperation.

From the above analysis, we can see that, as long as g_i is larger than a certain value, such as $(1-p_e)^{L_{max}} g_i > L_{max} c_i$, where L_{max} is a system parameter to indicate the maximum possible number of hops that a route is allowed to have, then varying g_i will not change the strategy design.

Until now, we have mainly focused on the situation that the game will be played for infinite duration. In most situations, a node will only stay in the network for finite duration. Then, for each player i, if $D_i^{max}(j)$ is too large, it may have helped its opponents much more than its opponents have helped it, while, if $D_i^{max}(j)$ is too small, it may lack enough nodes to forward packets for it. How to select a good $D_i^{max}(j)$ still remains as a challenge. Section 4 has studied the trade-off between the value of $D_i^{max}(j)$ and the performance through simulations, which shows that, under given simulation scenarios, a relatively small $D_i^{max}(j)$ value is good enough to achieve near-optimal performance (compared to setting $D_i^{max}(j)$ to be ∞) and good fairness (comparing to absolute fairness in cost). Here, it is also worth pointing out that the optimality of the proposed strategies cannot be guaranteed in finite duration scenarios.

3.4 Strategy Analysis under Attacks

In this paper, we focus on the following two widely used attack models are considered: dropping packet attack and injecting traffic attack. To simplify our illustration, we assume that $c_i = c$ and $g_i = g$ for all $i \in N$. We first study dropping packet attack. By dropping other nodes' packets, attackers can decrease the network throughput and waste other nodes' limited resources. According to the proposed attacker detection strategy, from an attacker's point of view, dropping all packets may not be a good strategy since this can be easily detected. Intuitively, in order to maximize the damage, attackers should selectively drop some portion of packets to avoid being detected. According to the multinode attack-resistant and cheat-proof cooperation strategy, the maximum number of packets that an attacker can drop without being detected is upper-bounded by $np_e + x\sqrt{np_e(1-p_e)}$, where n is the number of times that it has agreed to forward. That is, it has to forward at least $n(1-p_e) - x\sqrt{np_e(1-p_e)}$ packets. However, among those dropped packets, $n(1-p_e)$ packets can be caused by noise even if no attackers are present. Thus, the extra damage is upper-bounded by $x\sqrt{np_e(1-p_e)}c$. Meanwhile, the extra cost is $n(1-p_e)c - x\sqrt{np_e(1-p_e)}c$. Since, for any constant value $x \in \mathcal{R}^+$, we have

$$\lim_{n\to\infty} \frac{x\sqrt{np_e(1-p_e)}}{n(1-p_e)} = 0, \tag{22}$$

selectively dropping packets can bring no gain to the attackers. In other words, if the game will be played for an infinite duration, dropping packet attack cannot cause damage to selfish nodes.

Now, we study injecting traffic attack. By injecting an overwhelming number of packets to the network, attackers can consume other nodes' resources when they help the attacker's forwarding packets. Since, for each selfish node $i \in N_s$, we have $D_i(j,t) \leq D_i^{max}(j,t)$, the maximum number of packets that an attacker j can request i to forward without paying back is upper-bounded by $D_i^{max}(j,t)$. Therefore, the damage that can be caused by injecting traffic attack is bounded and limited and becomes negligible when $\lim_{t\to\infty} \frac{D_i^{max}(j,t)}{T_i(t)} = 0$.

In summary, when the multinode attack-resistant and cheat-proof strategy is used by all selfish nodes, attackers can only caused limited damage to the network. Further, the relative damage will go to 0 when the game will be played for infinite duration. Since $R_i(j,t)$ and $F_j(i,t)$ are in the same order, for any constant value $x \in \mathcal{R}^+$, we always have

$$\lim_{R_i(j,t)\to\infty} \frac{x\sqrt{R_i(j,t)p(1-p)}}{F_j(i,t)} = 0. \tag{23}$$

Therefore, except some false positive, selfish players' overall payoff will not be affected under attacks. Although false positive may cause a node unable to get help from some other nodes, this will not become a big issue since the false positive probability can be made to approach 0 by using a large constant x without decreasing the overall payoff. From the above analysis, we can also see that no matter what objectives the attackers have and what attacking strategy they use, as long as selfish nodes apply the multinode attack-resistant and cheat-proof cooperation strategy, the selfish nodes' performance can be guaranteed.

Based on the above analysis, we can conclude that, for the infinite duration case, an attacker j's overall payoff is upper-bounded by

$$U_j^m \le \lim_{t\to\infty} \sum_{i\in N_s} \frac{D_i^{max}(j,t)}{t} c, \tag{24}$$

provided that all selfish nodes follow the multinode attack-resistant and cheat-proof cooperation strategy. This upper-bound can be achieved by the following attacking strategy, which we refer to as **Optimal Attacking Strategy**: *In the secure routing and packet forwarding game, for any attacker $j \in N_m$, it should always refuse in the route participation stage, should always pick the route including no attackers in the route selection stage, and should not forward packets in the packet forwarding stage.*

Following similar arguments as in the proof of Theorem 1, we can also show that, in the infinite duration secure routing and packet forwarding game, the strategy profile where all selfish players follow the multinode attack-resistant and cheat-proof cooperation strategy and all attackers follow the above optimal attacking strategy forms a subgame perfect equilibrium, is cheat-proof and strongly Pareto optimal, and achieves absolute fairness in cost under some mild conditions.

When the game will only be played for finite duration, the above attacking strategy is not optimal any more. Now, the attackers can try to drop some nodes' packets without being detected, since the statistical dropping packet attacker detection will not be initiated unless having collected enough interactions (i.e., $R_i(j,t)$ is large enough in (19)) to avoid high false positive probability. In this case, selfish nodes' performance will be degraded a little bit. However, as long as the game will be played for a reasonably long time, which is the focus of this paper, the relative damage is still insignificant.

3.5 Discussion

Compared with existing work, such as [11], [12], [13], [14], [15], [16], we have addressed cooperation stimulation under more realistic and challenging scenarios: noisy environment, existence of (insider) attackers, variable traffic rate, etc. In such scenarios, instead of enforcing all nodes to act fully cooperatively, our goal is to stimulate cooperation among nodes as much as possible.

One major difference between our scheme and most existing reputation-based schemes is that, in our scheme, pairwise relationships have been maintained by nodes. That is, each selfish node will keep track of the interactions with all the other nodes met by it. The drawback is that it requires per-node monitoring and will result in extra storage overhead. However, the advantage lies in that it can effectively stimulate cooperation in noisy and hostile environments. Actually, for each node i, at any time, it only needs to maintain the records $R_i(j)$, $isBad_i(j)$, $F_i(j)$, and $F_j(i)$ for any other node j that i has interacted with, so the maximum storage overhead is upper-bounded by $4|N|$. As long as $|N|$ is not too large, for most mobile devices, such as notebook and PDA, the storage requirement is insignificant.

In most existing work, such as in [11], [12], [15], [16], however, each node makes its decision based solely on its own experienced quality of service, such as throughput. Although the overhead is much lower than our scheme due to that only end-to-end acknowledge is required and each node only needs to keep its own past state, they cannot effectively stimulate cooperation at all in noisy and hostile environments. As explained in Section 3.2, their logic is that no node will behave maliciously since misbehaviors will be propagated back later and the quality of service experienced by the misbehaving nodes will also be decreased. However, such logic cannot hold in noisy and hostile environments. First, attackers will be willing to see such performance degradation; therefore, they will try to behave maliciously if possible. Second, even only noise itself can cause such misbehavior propagation and performance degradation since noise can cause packet dropping. Meanwhile, without per-node monitoring, attackers can always behave maliciously and cause damage to the others without being detected.

In [11], [16], when a node makes its cooperation decision at each step, it only bases on the normalized throughput that it has experienced. If only normalized throughput is used, a greedy user can set a low forwarding ratio, but try to send a lot of packets. Therefore, unless the others also try to send a lot of packets, from the greedy node's point of view, even after a large portion of its packets have been dropped by other nodes, it can still enjoy a high throughput, although the normalized throughput may be low. Meanwhile, as mentioned before, applying the same forwarding ratio to all nodes is not fair to those who have acted cooperatively. To resolve this problem, in our scheme, each node applies a different packet forwarding decision for different nodes based on its past interactions between them.

In addition to reputation-based cooperation stimulation schemes, pricing-based schemes have also been proposed in the literature, such as [2], [3], [4], [5], [6]. Comparing to pricing-based schemes, the major drawback of reputation-based schemes is that some nodes may not get enough help to send out all their packets. The most underlying reasons are that favors cannot be returned immediately and the future is not predictable. In other words, when a node is requested by another node to forward packets, since it cannot get compensation immediately and it is not sure whether the requester will return the favor later, it usually has no strong incentive to accept the request. Pricing-based schemes do not suffer such problems since a node can get immediate

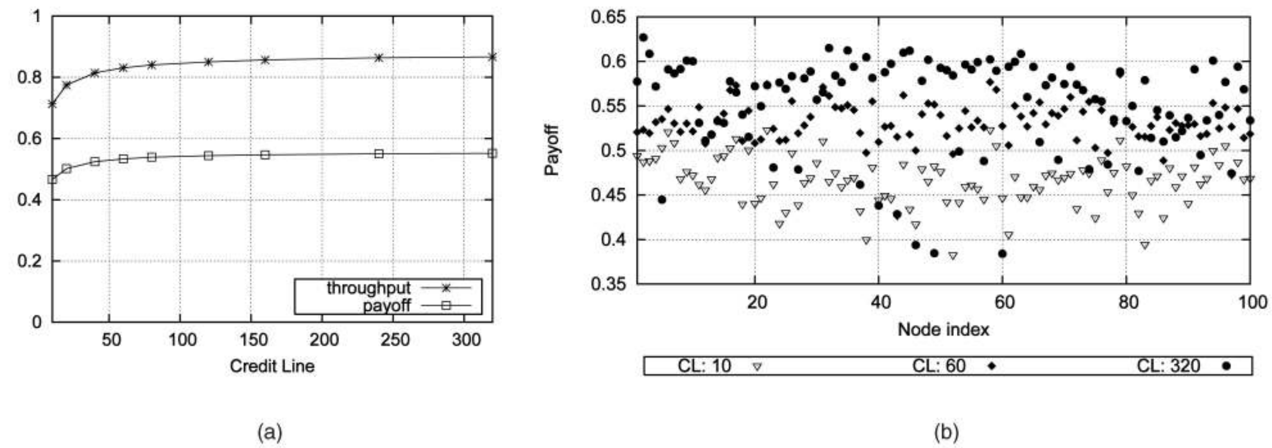


Fig. 2. Selfish nodes' performance under the proposed strategies.

monetary payback after providing services. However, pricing-based schemes require tamper-proof hardware or central banking service to handle billing information, which is their major drawback. If such a requirement can be efficiently satisfied with low overhead, pricing-based schemes can be a better choice than reputation-based schemes. Meanwhile, it is worth pointing out that pricing-based schemes also suffer from noise and possible malicious behavior, and the proposed statistical attacker detection mechanism is also applicable to pricing-based scenarios.

In general, necessary monitoring is needed when stimulating cooperation among nodes. For example, in [7], watchdog is proposed to detect whether some nodes have dropped packets. In this paper, we assume that the underlying monitoring mechanism can provide accurate per-node monitoring. Although this can be a strong assumption in some scenarios, it can greatly simplify our analysis and, at the same time, provide thoughtful insights. It is worth mentioning that, in some situations, perfect monitoring is either not available or too expensive to afford. Meanwhile, in our current analysis, the cost incurred by the underlying monitoring mechanism has not been included. The study of imperfect monitoring is beyond the scope of this paper, but will be investigated in our future work.

In our analysis, we have assumed that the packet drop ratio p_e is the same for all nodes at all times, which may not hold in general. If different nodes may experience different p_e, the nodes experiencing lower p_e may experience high false positive probabilities when performing the proposed attacker detection mechanism. In this case, to decrease false positive probability, nodes need set the threshold to be large enough, that is, using a larger x and p_e in (19). Although this may be taken advantage of by the attackers to cause more damage, as long as the gap between the packet dropping ratios experienced by different nodes is not large, which is usually the case, the extra damage is still limited.

It is also worth mentioning that the security of the proposed strategy also relies on the existing secure protocols such as those in [10], [18], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]. In general, besides dropping packets and injecting traffic, attackers can also have a variety of ways to attack the network, such as jamming, slander, etc. Instead of trying to address all these attacks, in this paper, our goal is to provide insight on stimulating cooperation in noisy and hostile environments.

4 Simulation Studies

We have conducted a set of simulations to evaluate the performance of the proposed strategies under various scenarios. In these simulations, 100 selfish nodes and various numbers of attackers are randomly deployed inside a rectangular area of $1,000 \text{ m} \times 1,000 \text{ m}$. Each node may either be static or move according to the *random waypoint* model [31]: A node starts at a random position, waits for a duration called the *pause time*, then randomly chooses a new location and moves toward the new location with a velocity uniformly chosen between v_{min} and v_{max}. The physical layer assumes that two nodes can directly communicate with each other successfully only if they are in each other's transmission range, which is set to be 250 m. The MAC layer protocol simulates the IEEE 802.11 DCF with a four-way handshaking mechanism [32]. The link bandwidth is 4 Mbps and the data packet size is 1,024 bytes. DSR [33] is used as the underlying routing protocol. For each simulation, each node randomly picks another node as the destination to send packets according to a Poisson random process. The average packet interarrival time is 1 s for each selfish node and 0.2 s for each attacker. Meanwhile, when a packet is dropped, no retransmission will be applied. We set $g_i = 1$, $c_i = 0.1$, and $t_f = 15{,}000 \text{ s}$.

4.1 Simulation Studies with Different Credit Lines

We first study how different credit lines can affect cooperation stimulation. In this set of simulations, there are only mobile selfish nodes which follow the multinode attack-resistant and cheat-proof cooperation strategy. Each mobile node follows the random waypoint mobility pattern with $v_{min} = 10 \text{ m/s}$, $v_{max} = 30 \text{ m/s}$, and pause time 100 s. In each simulation, the credit line (CL) is fixed to the same for all selfish nodes. The simulation results are depicted in Fig. 2. Fig. 2a demonstrates the relationship between CL, the average payoff, and the normalized throughput of selfish nodes. From these results, we can see that, when the credit line is over 80, the selfish nodes' average payoff does not increase any more, though the throughput may still

TABLE 1
The Four Ad Hoc Networks Considered

Mobile network 1:	All nodes follow random waypoint pattern with $v_{min} = 10m/s$, $v_{max} = 30m/s$, and pause time 100s.
Mobile network 2:	All nodes follow random waypoint pattern with $v_{min} = 10\alpha m/s$, $v_{max} = 30\alpha m/s$, and pause time 100s, where α is a value randomly drawn in the range [0, 1].
Partially mobile network:	The first 50 nodes follow random waypoint pattern with $v_{min} = 10m/s$, $v_{max} = 30m/s$, and pause time 100s, while the second 50 nodes are static.
Static network:	All nodes are static.

increase a little bit. This suggests that setting $CL = 80$ is almost optimal.

Now, we examine each individual node's payoff. Fig. 2b shows the individual nodes' payoff under three CL settings. First, we can see that, for each node, setting $CL = 60$ results in much higher payoff than setting $CL = 10$. Second, although, on average, setting $CL = 320$ can result in a slightly higher payoff than setting $CL = 60$, it is also easy to notice that a large portion of nodes (more than 20 percent) suffer much lower payoff than setting $CL = 60$. The reason is that these nodes have acted too generously in the sense that they have forwarded many more packets for the others than the others have done for it, which consequently decreases their payoff. This suggests that, when a node will only stay in the network for a finite duration, it may not be a good choice to set a very high credit line. In the following simulations, each selfish node will set $CL = 60$.

4.2 Simulation Studies under Different Networks

In this section, we investigate the performance of the proposed cooperation strategies in different ad hoc networks. We still consider only selfish nodes where all nodes follow the multinode attack-resistant and cheat-proof cooperation strategy. However, instead of considering only mobile ad hoc networks, we have also conducted simulations in static and partially mobile ad hoc networks. Specifically, the following four types of ad hoc networks are considered in Table 1.

Fig. 3 depicts each individual node's payoff under each of the four network settings. First, we can see that most nodes have very low payoff in the static ad hoc network. This is easy to understand: After these nodes have used up all the credit lines assigned by their neighbors to them, and since their neighbors may not need their help due to topology dependence, these nodes cannot request that their neighbors forward packets for them any more. Second, we can also see that some nodes in a static network still have very high payoff (almost 0.9). This is becausem for these nodes, their destinations are within their one-hop transmission range, so they do not rely on the others to forward packets.

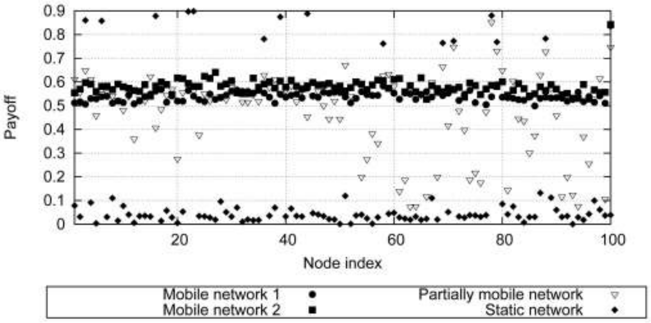


Fig. 3. Performance comparison in different networks.

What if some nodes are mobile in the network? Now, let us examine selfish nodes' performance under partially mobile ad hoc networks. First, the 50 mobile nodes (the nodes with index between 1 and 50) have almost comparable payoff to nodes in mobile ad hoc networks, though several of them still have a little bit lower payoff. Second, for the 50 static nodes (with index between 51 and 100), the majority of them suffer fairly low payoff, though some of them have high payoff because their destinations are within their transmission ranges. Third, comparing to the static ad hoc network, even those 50 static nodes have much higher payoff. These results suggest that mobility can play a very positive role in alleviating the effect of topology dependency.

Now, let us come back to mobile ad hoc networks. First, from Fig. 3, we can see that, in both mobile ad hoc networks (Mobile network 1 and 2), all nodes experience fairly high payoff, and all payoffs lie in a narrow range. These results suggest that the proposed strategies can effectively simulate cooperation among selfish nodes in mobile ad hoc networks. Second, nodes in mobile network 2 have slightly higher payoff than nodes in mobile network 1. This is because mobile network 2 has a lower mobility rate than mobile network 1, which leads to low link breakage ratio.

It is worth pointing out that, according to the random waypoint model, each mobile node can move globally inside the specified area. Sometimes nodes' movement may be restricted in a certain local area. In the future, we will also investigate how restricted movement can affect cooperation stimulation. Actually, the above partially mobile network can be regarded as a special case of such network with restricted movement since half nodes of this network are restricted to fixed locations.

4.3 Simulation Studies under Attacks

Now, we study how the proposed cooperation stimulation strategies can effectively handle attacks, specifically, dropping packets attack and injecting traffic attacks. In this set of simulations, selfish nodes follow the multinode attack-resistant and cheat-proof cooperation strategy and attackers follow the optimal attacking strategy. All nodes are mobile, and each follows the random waypoint mobility pattern with $v_{min} = 10$ m/s, $v_{max} = 30$ m/s, and pause time 100s. For each selfish node, the maximum allowable false positive probability is set to be 0.1 percent, and the link breakage ratio p_e is estimated based on its own experience, which is the ratio between the total number of link breakages it has experienced with itself being the transmitter and the total number of transmissions it has tried. Fig. 4 draws the estimated values of p_e under the current

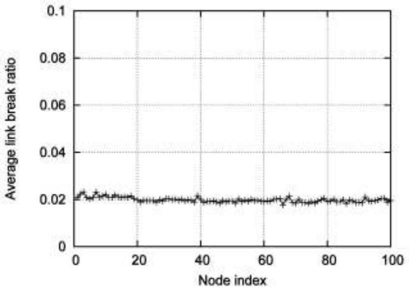


Fig. 4. Estimated link breakage ratio.

network configuration, which shows that all nodes have almost the same link breakage ratio (here, 2 percent).

We first study how well the proposed strategies can scale with the increased number of attackers. Fig. 5a shows the selfish nodes' average payoff under various number of attackers, and Fig. 5b shows the total damage that the attackers have caused to selfish nodes, where total damage is defined in (13) without being divided by t_f. First, from Fig. 5b, we can see that the total damage that the attackers can cause increases almost linearly with the increase of attacker number. This makes sense since, the higher the number of attackers, the more traffic they can inject and the more packets they can drop. Second, from Fig. 5a, we can see that the selfish nodes' average payoff decreases only very slightly with the increased number of attackers. Further, in the above simulation results, there also exist some points where the selfish nodes' payoff may even increase a little bit with the increased number of attackers. Such behavior can be better illustrated by Fig. 5c, where, in this figure, we have drawn the selfish nodes' average payoff with the increase of time under both no attackers and under 20 attackers. From Fig. 5c we can see that, in the first 4,000 s, the selfish nodes can have higher payoff under no attackers than under 20 attackers, while, later, the selfish nodes' payoff under attacks may even outperform the payoff under no attacks. This phenomenon can be explained as follows: Initially, the damage is mainly caused by injecting traffic attacks, then, after attackers have used up all the credit lines assigned by the selfish nodes, the damage is contributed mostly by dropping packet attacks. However, since the link breakage ratio is low, the attackers can only drop very few packets without being detected. The reasons the selfish nodes' payoff under attacks may even outperform the payoff under no attacks come from 1) the randomness of each simulation and 2) because, by participating packet forwarding, the attackers can also decrease the average number of hops per selected route, which consequently increases the selfish nodes' payoff.

In the final set of simulations, we demonstrate why setting a very high credit line may not be a good choice. In this set of simulations, we fix the number of attackers to be 20, but vary the credit lines in each simulation, ranging from 20 to 100. The simulation results are illustrated in Fig. 6. From Fig. 6b, we can see that, with the increase of credit line, the damage that can be caused by attackers will also increase linearly because attackers can inject more packets. From Fig. 6a, we can see that increasing the credit line may even decrease the selfish nodes' performance. For example, the selfish nodes with $CL = 100$ have even lower payoff than selfish nodes with $CL = 80$. This can be easily understood by examining the results presented in Fig. 6b: Setting the credit line to be 100 will let the selfish nodes suffer more damage than setting the credit line to be 80.

5 Conclusion

In this paper, we have formally investigated secure cooperation stimulation in autonomous mobile ad hoc

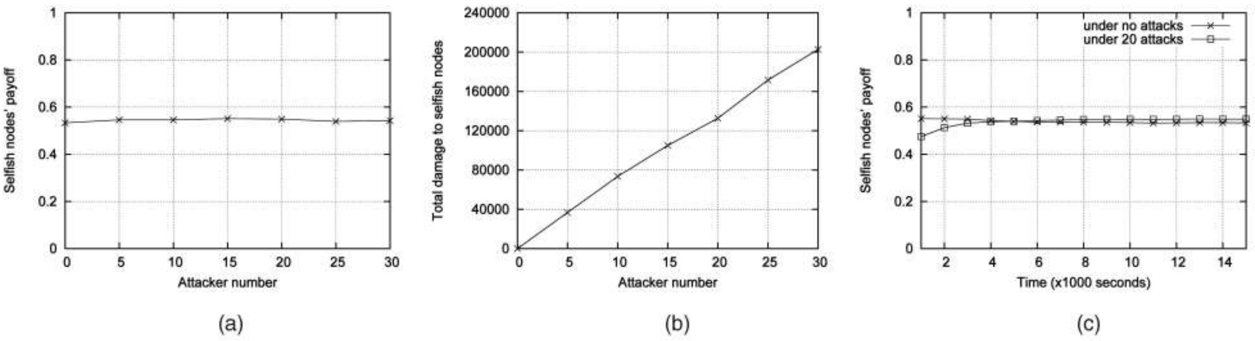


Fig. 5. Performance study under attacks.

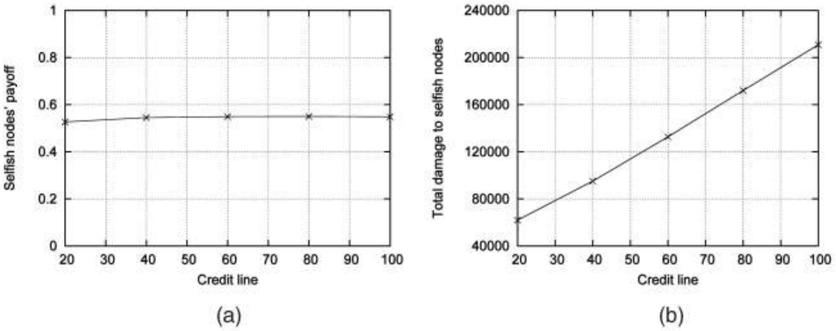
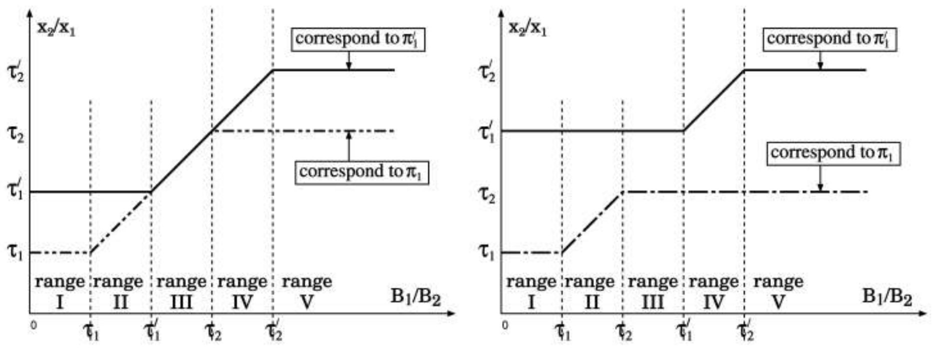


Fig. 6. The effect of credit lines under attacks.



(a)

(b)

Fig. 7. Player 1 falsely reports the value of π_1. (a) $\tau_1' < \tau_2$. (b) $\tau_1' > \tau_2$.

networks under a game theoretic framework. Besides selfish behavior, possible attacks have also been studied, and attack-resistant cooperation stimulations have been devised which can work well under noisy and hostile environments. First, a simple yet illuminating two-player packet forwarding game is studied. To find good cooperation strategies, equilibrium refinements have been performed on obtained Nash equilibrium solutions under different optimality criteria, including subgame perfection, Pareto optimality, fairness, and cheat-proofing. Finally, a unique Nash equilibrium solution is derived, which states that, in the two-node packet forwarding game, a node should not help its opponent more than its opponent has helped it.

The results are then extended to handle multinode scenarios in noisy and hostile environments, where the dynamic interactions between nodes are modeled as secure routing and packet forwarding games. By taking into consideration the difference between two-node case and multinode case, an attack-resistant and cheat-proof cooperation stimulation strategy has been devised for autonomous mobile ad hoc networks. The analysis has demonstrated the effectiveness of the proposed strategy and shown that it is optimal under certain conditions. The analysis has also shown that the damage that can be caused by attackers is bounded and limited when the proposed strategies are used by selfish nodes. Simulation results have also illustrated that the proposed strategies can effectively stimulate cooperation among selfish nodes in noisy and hostile environments.

Appendix

Cheat-Proofness Analysis for Solution (7), (8), and (9)

We first study the solution (7). Let $\pi_i = c_i/g_i$ denote player i's cost-gain (CG) ratio. We first analyze whether player i can increase its payoff by reporting a false CG ratio given that player 2 will honestly report its CG value. That is, we fix the value of π_2, letting π_1 be player 1's true value, and letting π_1' be the value that player 1 will falsely report with $\pi_1' > \pi_1$. Let

$$\tau_1 = \frac{2}{\pi_2 + \frac{1}{\pi_1}},\ \tau_2 = \frac{\pi_1 + \frac{1}{\pi_2}}{2},\ \tau_1' = \frac{2}{\pi_2 + \frac{1}{\pi_1'}},\ \text{and}\ \tau_2' = \frac{\pi_1' + \frac{1}{\pi_2}}{2}.$$

It is easy to check that $\tau_1 < \tau_1'$ and $\tau_2 < \tau_2'$. Recall that B_i is the maximum number of packets that player i will request its opponent to forward for it in each stage. Let (x_1, x_2) denote the number of packets in average they will forward for each other in each stage according to the solution (7) given that the true values of B_1 and B_2 are known by both players. The relationship between x_2/x_1 and B_1/B_2 under different situations is illustrated in Fig. 7a and Fig. 7b. In these two figures, the dashed curve corresponds to the relationship between x_2/x_1 and B_1/B_2 given that player 1 honestly reports its CG value, which is π_1, while the solid curve corresponds to the relationship between x_2/x_1 and B_1/B_2 given that player 1 falsely reports its CG value, which is π_1'.

From the results illustrated in Fig. 7, we can see that, by falsely reporting a higher CG ratio, in most situations, player 1 can increase the ratio of x_2/x_1. Next, we study the effect of falsely reporting a high CG ratio on player 1's payoff. We first consider the situation that $\tau_1' \leq \tau_2$, which is illustrated in Fig. 7a. In this case, the whole feasible space can be partitioned into five subareas along the feasible range of B_1/B_2:

- For any value of B_1/B_2 inside range I, the solution corresponding to π_1 is

  $$\left(\frac{\pi_2 + \frac{1}{\pi_1}}{2} B_1, B_1\right)$$

  and the solution corresponding to π_1' is

  $$\left(\frac{\pi_2 + \frac{1}{\pi_1'}}{2} B_1, B_1\right).$$

  Since $\pi_1' > \pi_1$, by falsely reporting a higher CG ratio, player 1 can forward fewer packets for player 2 than it should, consequently increasing its own payoff.
- For any value of B_1/B_2 inside range II, the solution corresponding to π_1 is (B_2, B_1) and the solution corresponding to π_1' is

  $$\left(\frac{\pi_2 + \frac{1}{\pi_1'}}{2} B_1, B_1\right).$$

  Since $B_2 > \frac{\pi_2 + \frac{1}{\pi_1'}}{2} B_1$, by falsely reporting a higher CG ratio, player 1 can forward fewer packets for player 2 than it should, consequently increasing its own payoff.

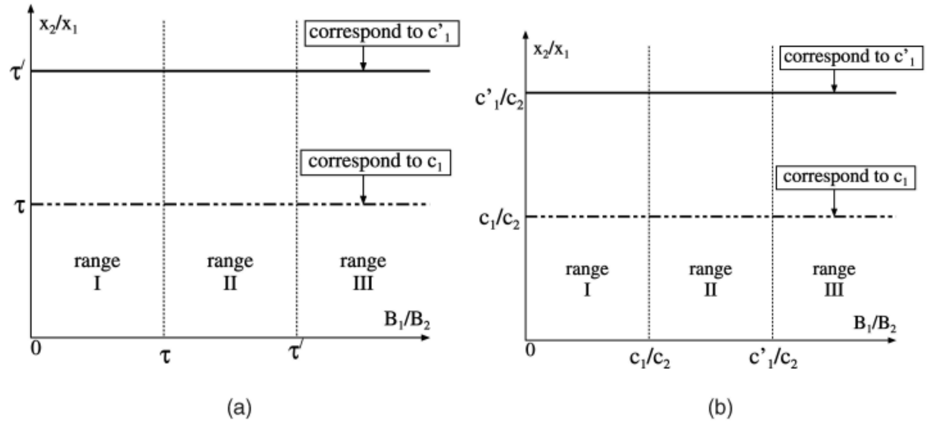


Fig. 8. Player 1 falsely reports its cost.

- For any value of B_1/B_2 inside range III, the solution corresponding to π_1 is (B_2, B_1) and the solution corresponding to π_1' is also (B_2, B_1). That is, in this situation, by changing the value of π_1 to π_1', player 1's payoff will not change.
- For any value of B_1/B_2 inside range IV, the solution corresponding to π_1 is
$$\left(B_2, \frac{\pi_1 + \frac{1}{\pi_2}}{2} B_2\right)$$
and the solution corresponding to π_1' is (B_2, B_1). Since $B_1 > \frac{\pi_1 + \frac{1}{\pi_2}}{2} B_2$, by falsely reporting a higher CG ratio, player 1 can request player 2 to forward more packets for it than player 2 should, consequently increasing its own payoff.
- For any value of B_1/B_2 inside range V, the solution corresponding to π_1 is
$$\left(B_2, \frac{\pi_1 + \frac{1}{\pi_2}}{2} B_2\right)$$
and the solution corresponding to π_1' is
$$\left(B_2, \frac{\pi_1' + \frac{1}{\pi_2}}{2} B_2\right).$$
Since $\frac{\pi_1' + \frac{1}{\pi_2}}{2} B_2 > \frac{\pi_1 + \frac{1}{\pi_2}}{2} B_2$, by falsely reporting a higher CG ratio, player 1 can request player 2 to forward more packets for it than player 2 should, consequently increasing its own payoff.

Similar results can also be obtained for the case that $\tau_1' > \tau_2$, where player 1 can now increase its own payoff over all possible values of B_1/B_2 by falsely reporting a higher π_1 value given that π_2 is fixed. In summary, by falsely reporting a higher π_1 value, in most situations, player 1 can increase its payoff and, in no situations, player 1's payoff will be decreased. Further, the higher player 1 reports the value of the CG ratio, the more benefit player 1 can get. Similarly, player 2 can also increase its benefit by falsely reporting a higher CG ratio.

Next, we consider the solution (8). Now, let $\tau = \frac{g_2 + c_1}{g_1 + c_2}$ and $\tau' = \frac{g_2 + c_1'}{g_1 + c_2}$, where $c_1 < c_1'$. Fig. 8a illustrates the relationship between x_2/x_1 and B_1/B_2 for the two different reported cost values c_1 and c_1', where g_1, g_2, and c_2 are fixed. Similarly as in Fig. 7, the dashed curve corresponds to the case that player 1 reports a true cost value, while the solid curve corresponds to the case that player 1 reports a false cost value. From Fig. 8a, we can see that, by falsely reporting a higher cost value, in all situations, player 1 can increase the ratio of x_2/x_1. Next, we study the effect of falsely reporting a higher cost on player 1's payoff. As shown in Fig. 8a, the whole space can be partitioned into three subareas along the feasible range of B_1/B_2:

- For any value of B_1/B_2 inside range I, the solution corresponding to c_1 is $(B_1/\tau, B_1)$ and the solution corresponding to c_1' is $(B_1/\tau', B_1)$. Since $\tau' > \tau$, by falsely reporting a higher cost, player 1 can forward fewer packets for player 2 than it should, thus increasing its own payoff.
- For any value of B_1/B_2 inside range II, the solution corresponding to c_1 is $(B_2, \tau B_2)$ and the solution corresponding to c_1' is $(B_1/\tau', B_1)$. Since $B_1/\tau' < B_2$ and $B_1 > \tau B_2$, by falsely reporting a higher cost, player 1 can forward fewer packets for player 2 than it should and request player 2 to forward more packets for it than player 2 should, thus increasing its own payoff.
- For any value of B_1/B_2 inside range III, the solution corresponding to c_1 is $(B_2, \tau B_2)$ and the solution corresponding to c_1' is $(B_2, \tau' B_2)$. Since $\tau' > \tau$, by falsely reporting a higher cost, player 1 can always request player 2 to forward more packets for it than player 2 should do, thus increasing its own payoff.

In summary, by falsely reporting a higher c_1 value, in all situations, player 1 can increase its payoff given that c_2 and g_2 are fixed. Further, the higher player 1 reports the value of c_1, the higher the benefit that player 1 can get. Applying similar analysis, it is also easy to show that, by falsely reporting a lower g_1 value, in all situations, player 1 can increase its payoff given that c_2 and g_2 are fixed. Similarly, player 2 can also increase its benefit by falsely reporting a higher c_2 or a lower g_2 given that g_1 and c_1 are fixed.

Now, we consider the solution (9). Fig. 8b illustrates the relationship between x_2/x_1 and B_1/B_2 for the two different reported cost values c_1 and c_1' with $c_1' > c_1$ and

c_2 fixed. From Fig. 8b, we can see that, by falsely reporting a higher c_1 value, in all situations, player 1 can increase the ratio of x_2/x_1.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and feedback. This work was supported in part by the US Army Research Office under URI Award No. DAAD19-01-1-0494.

References

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

[2] L. Buttyan and J.P. Hubaux, "Enforcing Service Availability in Mobile Ad-Hoc Networks," *Proc. ACM MobiHoc,* 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," *Proc. IEEE INFOCOM,* 2003.

[5] L. Anderegg and S. Eidenbenz, "Ad Hoc-VCG: A Truthful and Cost-Efficient Routing Protocol for Mobile Ad Hoc Networks with Selfish Agents," *Proc. ACM MobiCom,* 2003.

[6] S. Zhong, L. Li, Y.G. Liu, and Y.R. Yang, "On Designing Incentive-Compatible Routing and Forwarding Protocols in Wireless Ad-Hoc Networks—An Integrated Approach Using Game Theoretical and Cryptographic Techniques," *Proc. ACM MobiCom,* pp. 117-131, 2005.

[7] S. Marti, T.J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *Proc. ACM MobiCom,* 2000.

[8] P. Michiardi and R. Molva, "Core: A COllaborative REputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," *Proc. IFIP—Comm. and Multimedia Security Conf.,* 2002.

[9] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the CONFIDANT Protocol," *Proc. ACM MobiHoc,* 2002.

[10] W. Yu and K.J.R. Liu, "Attack-Resistant Cooperation Stimulation in Autonomous Ad Hoc Networks," *IEEE J. Selected Areas in Comm.,* vol. 23, no. 12, pp. 2260-2271, Dec. 2005.

[11] V. Srinivasan, P. Nuggehalli, C.F. Chiasserini, and R.R. Rao, "Cooperation in Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM,* 2003.

[12] A. Urpi, M. Bonuccelli, and S. Giordano, "Modeling Cooperation in Mobile Ad Hoc Networks: A Formal Description of Selfishness," *Proc. Workshop Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOPT '03),* 2003.

[13] J. Crowcroft, R. Gibbens, F. Kelly, and S. Ostring, "Modelling Incentives for Collaboration in Mobile Ad Hoc Networks," *Proc. Workshop Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT '03),* 2003.

[14] P. Michiardi and R. Molva, "A Game Theoretical Approach to Evaluate Cooperation Enforcement Mechanisms in Mobile Ad Hoc Networks," *Proc. Workshop Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT '03),* 2003.

[15] E. Altman, A.A. Kherani, P. Michiardi, and R. Molva, "Non-Cooperative Forwarding in Ad-Hoc Networks," technical report, INRIA, Sophia Antipolis, 2004.

[16] M. Felegyhazi, J.-P. Hubaux, and L. Buttyan, "Nash Equilibria of Packet Forwarding Strategies in Wireless Ad Hoc Networks," *IEEE Trans. Mobile Computing,* vol. 5, no. 5, pp. 463-476, Sept.-Oct. 2006.

[17] M.J. Osborne and A. Rubinste, *A Course in Game Theory.* The MIT Press, 1994.

[18] W. Yu, Y. Sun, and K.J.R. Liu, "HADOF: Defense against Routing Disruptions in Mobile Ad Hoc Networks," *Proc. IEEE INFOCOM,* 2005.

[19] R. Dawkins, *The Selfish Gene,* second ed. Oxford Univ. Press, 1990.

[20] O. Kallenberg, *Foundations of Modern Probability.* Springer-Verlag, 1977.

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

[22] J.P. Hubaux, L. Buttyan, and S. Capkun, "The Quest for Security in Mobile Ad Hoc Networks," *Proc. ACM MobiHoc,* 2001.

[23] Y.-C. Hu, A. Perrig, and D.B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," *Proc. ACM MobiCom,* 2002.

[24] P. Papadimitratos and Z. Haas, "Secure Routing for Mobile Ad Hoc Networks," *Proc. SCS Comm. Networks and Distributed Systems Modeling and Simulation Conf. (CNDS '02),* Jan. 2002.

[25] K. Sanzgiri, B. Dahill, B.N. Levine, C. Shields, and E.M. Belding-Royer, "A Secure Routing Protocol for Ad Hoc Networks," *Proc. Int'l Conf. Network Protocols (ICNP '02),* Nov. 2002.

[26] M.G. Zapata and N. Asokan, "Securing Ad Hoc Routing Protocols," *Proc. Int'l Conf. Web Information Systems Eng.,* 2002.

[27] Y.-C. Hu, A. Perrig, and D.B. Johnson, "Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols," *Proc. Int'l Conf. Web Information Systems Eng.,* 2003.

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

[29] Y.-C. Hu, A. Perrig, and D.B. Johnson, "SEAD: Secure Efficient Distance Vector Routing for Mobile Wireless Ad Hoc Networks," *Ad Hoc Networks J.,* vol. 1, pp. 175-192, 2003.

[30] W. Yu and K.J.R. Liu, "Secure Cooperative Mobile Ad Hoc Networks against Injecting Traffic Attacks," *Proc. IEEE Int'l Conf. Sensor and Ad Hoc Comm. and Networks (SECON '05),* 2005.

[31] J. Yoon, M. Liu, and B. Noble, "Sound Mobility Models," *Proc. ACM MobiCom,* 2003.

[32] *IEEE Standard 802.11-1007, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,* IEEE Computer Soc. LAN MAN Standards Committee, 1999.

[33] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks, Mobile Computing," *Mobile Computing,* pp. 153-181, 1996.

Wei Yu received the BS degree in computer science from the University of Science and Technology of China (USTC) in 2000, the MS degree in computer science from Washington University in St. Louis in 2002, and the PhD degree in electrical engineering from the University of Maryland in 2006. From August 2000 to May 2002, he was a graduate research assistant at Washington University in St. Louis. From September 2002 to July 2006, he was a graduate research assistant with the Communications and Signal Processing Laboratory and the Institute for Systems Research at the University of Maryland. He joined Microsoft Corporation in August 2006. His research interests include network security, wireless communications and networking, game theory, wireless multimedia, and pattern recognition.

K.J. Ray Liu (F'03) received the BS degree from National Taiwan University and the PhD degree from the University of California, Los Angeles, both in electrical engineering. He is currently a professor and associate chair of graduate studies and research in the Electrical and Computer Engineering Department at the University of Maryland, College Park. His research contributions encompass broad aspects of wireless communications and networking, information forensics and security, multimedia communications and signal processing, bioinformatics and biomedical imaging, and signal processing algorithms and architectures. He is the recipient of numerous honors and awards including best paper awards from the IEEE Signal Processing Society (twice), the IEEE Vehicular Technology Society, and EURASIP. He also received the EURASIP Meritorious Service Award and the US National Science Foundation Young Investigator Award and was named an IEEE Signal Processing Society Distinguished Lecturer. He also received various teaching and research recognitions from the University of Maryland, including the Distinguished Scholar-Teacher Award, the Poole and Kent Company Senior Faculty Teaching Award, and the Invention of the Year Award. He is the vice president of publications and on the board of governors of the IEEE Signal Processing Society. He was the editor-in-chief of the *IEEE Signal Processing Magazine* and the founding editor-in-chief of the *EURASIP Journal on Applied Signal Processing*. He is a fellow of the IEEE.