A Reputation-based Mechanism for Isolating Selfish Nodes in Ad Hoc Networks

M. Tamer Refaei, Vivek Srivastava, Luiz DaSilva
Bradley Department of Electrical and Computer Engineering
Virginia Tech
Alexandria, VA 22314
{mtamer, vivs, ldasilva}@vt.edu

Mohamed Eltoweissy
Department of Computer Science
Virginia Tech
Falls Church, VA 22043
toweissy@vt.edu

Abstract

For ad hoc networks to realize their potential in commercial deployments, it is important that they incorporate adequate security measures. Selfish behavior of autonomous network nodes could greatly disrupt network operation. Such behavior should be discouraged, detected, and isolated. In this paper, we propose a reputation-based mechanism to detect and isolate selfish nodes in an ad hoc network. The proposed mechanism allows a node to autonomously evaluate the "reputation" of its neighbors based on the completion of the requested service. The underlying principle is that when a node forwards a packet through one of its neighbors, it holds that neighbor responsible for the correct delivery of the packet to the destination. Our mechanism is efficient and immune to node collusion since, unlike most contemporary mechanisms for reputation-based trust, it does not depend on exchanging reputation information among nodes. We also explore various reputation functions and report on their effectiveness in isolating selfish nodes and reducing false positives. Our simulation results demonstrate that the choice of the reputation function greatly impacts performance and that the proposed mechanism, with a carefully selected function, is successful in isolating selfish nodes while maintaining false positives at a reasonably low level.

I. Introduction

Multi-hop communication in mobile ad hoc networks (MANETs) requires collaboration among nodes, which forward packets for one another. Most studies of ad hoc networks assume that nodes can be programmed to always perform this forwarding functionality. In commercial deployment of MANETs, however, some nodes may refuse to forward packets in order to conserve their limited resources (for example, energy), resulting in traffic disruption. Nodes exhibiting such behavior are termed selfish [10]. Selfishness is usually passive behavior. Additionally, malicious nodes may intentionally, and without concern about their own resources, attempt to disrupt network operations by mounting denial-of-service attacks or by actively degrading the network performance. For example, malicious nodes could disrupt routing operation by advertising non-existent routes or sub-optimal routes. Selfish and malicious behaviors are usually distinguished based on the node's intent. Network disruption is a side effect of the behavior of a selfish node, while disrupting the network is the intent of malicious nodes. One way to recognize and isolate such disruptive node behavior is through trust management mechanisms [1] [2] [3] [6].

In this work, we focus on detection and isolation of selfish nodes in ad hoc networks. We propose a reputation-based mechanism as a means of building trust among nodes. The mechanism relies on the principle that a node autonomously (i.e., without communicating with other neighboring nodes) evaluates its neighbors based on the completion of the requested service(s). We note that this principle, in general, can be applied to operations that involve cooperation among nodes in an ad hoc network. We introduce an application of this principle to the routing functionality such that nodes are rewarded or penalized based on their behavior during packet forwarding. It is to be noted that each node sees many different flows with varying routes over time. Consequently, the evaluation will not be greatly biased by individual flows; rather it seeks to identify a pattern of selfish behavior. This is illustrated in figure 1.

In the figure, two flows are being carried over routes SABXED and MABF. Suppose B starts to act selfishly, dropping all packets that it is expected to forward. Eventually, nodes upstream of B will notice that packets are not being delivered to their intended destinations. Node A will reduce the reputation index

Proceedings of the Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)
0-7695-2375-7/05 $20.00 © 2005 IEEE

it assigns to B (twice, due to B's effect on both flows); A's immediate neighbors will in turn reduce the reputation index they assign to A. Once B's reputation index falls below a certain threshold, this triggers new route discovery processes, and the flows are re-routed to SAMLD and MXEF, bypassing the selfish node.

Previously proposed reputation-based trust management schemes primarily rely on the monitoring of neighbors' transmissions and the exchange of reputation information among nodes [1] [2] [3]. Our protocol provides several advantages over such schemes for reputation establishment, including:

1. *Routing protocol independence*: Our mechanism is based on feedback from the destination and hence is independent of the choice of the ad hoc routing protocol.
2. *No communication overhead*: Sharing of reputation information introduces additional control overhead. Our mechanism introduces no such load.
3. *Directional transmission*: Unlike other schemes, our mechanism does not require nodes to monitor their neighbors' transmissions. Hence, our proposed scheme is robust to the use of directional antennas.
4. *Elimination of an overlay trust management system*: Other schemes require a trust overlay to evaluate the reliability of reputation information shared.

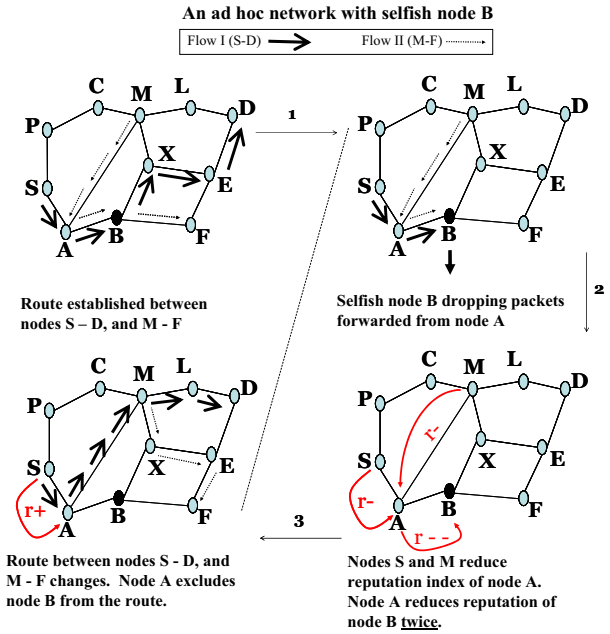

Figure 1. Operation of the protocol

The remainder of this paper is organized as follows. In Section II, we explain the main concept and features of our mechanism and discuss possible variations of the reputation-building component. In section III, we demonstrate through simulations the applicability of our mechanism to the fast isolation of selfish nodes. We also evaluate its effectiveness in maintaining a low percentage of false positives. We summarize related work in section IV and present our conclusions and directions for future work in section V.

II. Reputation-based Mechanism

A. *Description*

Our mechanism provides a distributed reputation evaluation scheme implemented autonomously at every node in an ad hoc network with the objective of identifying and isolating selfish neighbors. Each node maintains a reputation table, where a reputation index is stored for each of the node's immediate neighbors. Considering a network of nodes in $\mathbf{N}= \{1,2,\ldots,N\}$, r_{ij} denotes the reputation index of node j as assigned by node i, $\{\forall i, j \in N : d(i,j) = 1\}$, where $d(i,j)$ is the distance in hops between nodes i and j.

A node ascribes a reputation index to each of its neighbors based on successful delivery of packets forwarded through that neighbor. For each successfully delivered packet, each node along the route increases the reputation index of its next-hop neighbor that forwarded the packet. Conversely, packet delivery failures result in a penalty applied to such neighbors by decreasing their reputation index. In other words, when a node transmits a packet to one of its neighbors, it holds the neighbor responsible for the correct delivery of the packet to the final destination. The indication of a success or failure is obtained from feedback received from the destination (*e.g.*, using TCP acknowledgements). The function used to compute the reputation index is a design decision that is influenced by factors including node behavior, node location, as well as others. Later in this section, we present and evaluate three heuristics for the reputation function.

To prevent selfish behavior and to provide motivation for nodes to build up their reputation, each node determines whether to forward or drop a packet based on the reputation of the packet's previous hop. Once a node's reputation, as perceived by its neighbors, falls below a pre-determined threshold all

packets forwarded through or originating at that node are discarded by those neighbors and the node is isolated. Summarizing, the underlying approach is:

1. To evaluate neighboring nodes based on the completion of the requested tasks (packet delivery); and
2. To detect the completion of a task based on feedback received from the end host (delivery acknowledgement).

In this paper, the reputation threshold is assumed to be global (i.e., same value used by all nodes). Alternatively, it can be locally defined according to a node's preference. The significance and impact of locally defined threshold values will be explored in future research.

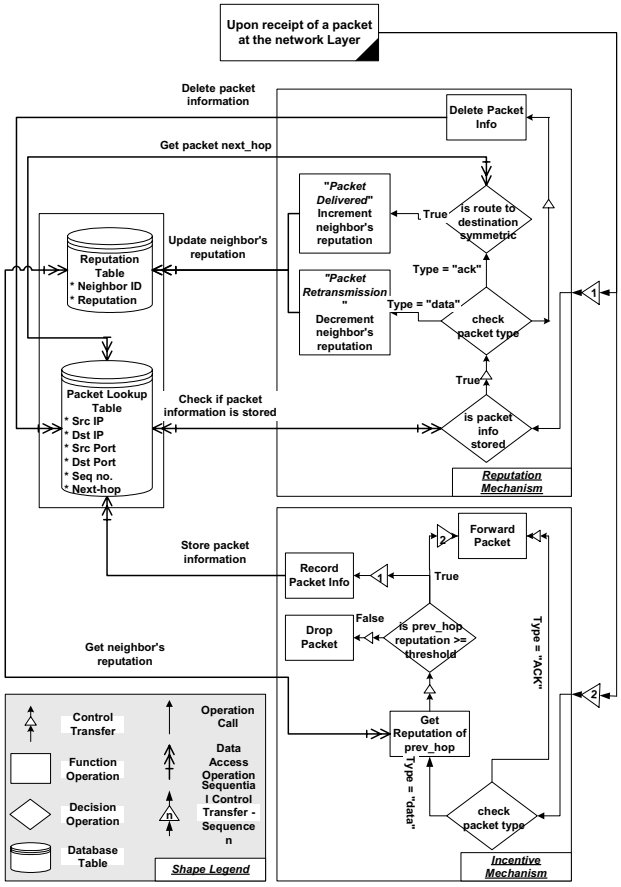

Figure 2. Flowchart illustration of our mechanism

B. *Operation*

We illustrate the operation of our mechanism in figure 2. Each node maintains a lookup table that stores information about data packets forwarded through it, including sequence numbers, source and destination IP addresses and port numbers, and the address of the next hop. Upon receiving a data packet, a node *i* checks whether the packet is a retransmission (indicated by the presence of stored packet information). If it is, node *i* decrements the reputation index of the neighbor through which the original packet was forwarded. Node *i* then compares the current reputation index of the previous hop *k* to a reputation threshold r_{thresh}. If $r_{ik} < r_{thresh}$, the packet is dropped. Otherwise, node *i* forwards the packet after storing its related information in the lookup table (creating an entry for new packets or updating entries for retransmitted packets). The size of the lookup table can be reduced by using hashing. Also, as packets age in the lookup table they can be expunged.

Upon receiving an acknowledgement from node *k*, node *i* verifies whether node *k* was the node through which the corresponding data packet was forwarded (recall that this information is stored in the lookup table). If so, node *i* increments r_{ik}, rewarding node *k* for the successful delivery of the data packet to the destination node. Hence, the algorithm only updates neighbors' reputation indices when the route between these nodes and the destination is symmetric.

The values by which the algorithm increments and decrements the reputation index of a node (what we call the reputation function) are important design parameters. The sharper the slope of the function, the faster the scheme manages to detect a selfish node. The tradeoff is that an overly aggressive scheme may also result in a higher number of false positives (note that false positives may result from packet losses due to reasons other than selfish node behavior, such as channel conditions, mobility or buffer overflow at intermediate nodes). We investigate three different reputation functions and derive conclusions on their applicability to different scenarios later in the paper.

Another design issue is the value of the reputation threshold. A detailed analysis to determine effective values of the threshold follows.

C. *Design Considerations*

1. *Parameters*

Each node maintains a reputation table that holds reputation information about the node's neighbors. Upon encountering a new neighbor *k*, node *i* creates an entry for that neighbor in its reputation table and initializes the reputation index to $r_{ik} = r_0$. A node updates the reputation of its neighbors based on the outcome of packet delivery events as discussed above.

Successful packet delivery events result in incrementing a neighbor's reputation index up to a maximum value $r_{\max}$, while failed packet delivery events result in decrementing the reputation index of the neighbor. A neighbor whose reputation falls below r_{thresh} is marked as selfish and is blacklisted. We note that $r_{\max} > r_0 > r_{thresh}$.

The choice of $\Delta_{\max} \equiv r_{\max} - r_0$ and $\Delta_{thresh} \equiv r_0 - r_{thresh}$ affects the sensitivity of our mechanism to packet drop events, which in turn affects the performance of the algorithm. An aggressive mode of operation corresponds to small values of $\Delta_{\max}$ and Δ_{thresh}, increasing sensitivity to packet drop events. This is likely to result in faster isolation of selfish nodes and a considerable number of false positives, where a node is falsely identified as selfish due to packet drop events unrelated to selfishness (such as congestion or collisions). Conversely, a more conservative mode of operation corresponds to larger values of $\Delta_{\max}$ and Δ_{thresh}, resulting in slower isolation of selfish nodes but also fewer false positives. We find that the values of $\Delta_{\max}$ and Δ_{thresh} should be selected based on factors such as network density, average network radius, traffic load, and expected number of selfish nodes.

Consider for example a 4x4 grid network topology where each node can establish direct links to its neighbors in the horizontal and vertical directions (but not diagonally); and two selfish nodes are present in the network (these nodes drop all packets received from their neighbors) We generate 80 CBR (constant bit rate) flows and set $r_0 = 50$. Averaged over 10 simulation runs, table 1 presents the percentage of selfish nodes that are successfully isolated (i.e. identified as selfish by all their neighbors) by the end of the simulation time and the percentage of false positives (i.e. nodes that are falsely identified as selfish) , for several choices of $\Delta_{\max}$ and Δ_{thresh}. Under these conditions, $\Delta_{\max} = 30$ and $\Delta_{thresh} = 15$ achieve a low rate of false positives while isolating selfish nodes reasonably fast.

We note that for a scenario where selfish nodes drop all
traffic forwarded through them, increasing the value of $\Delta_{\max}$ while keeping $\Delta_{thresh} = 15$ does not affect the results obtained for isolation. However, the results for false positives are improved. We use $\Delta_{\max} = 50$ and $\Delta_{thresh} = 15$ for all simulations discussed in the remainder of the paper.

Table 1.
Comparison of different values of $\Delta_{\max}$ and Δ_{thresh} for $r_0 = 50$. Percentage of selfish nodes isolated and false positives are averaged over 10, 600-sec. simulations.

D.thresh / D. max	5	15	25
10	-Isolation: 100% at 208.5 -FalsePos: 6.1%	-Isolation: 85% at 220.5 -FalsePos: 3.2%	-Isolation: 75% at 335 -FalsePos: 1.8%
20	-Isolation: 95% at 163.5 -FalsePos: 3.7%	-Isolation: 90% at 250 -FalsePos: 1.8%	-Isolation: 70% at 395 -FalsePos: 0.9%
30	-Isolation: 95% at 153 -FalsePos: 3.4%	-Isolation: 85% at 232.5 -FalsePos: 1.1%	-Isolation: 65% at 445 -FalsePos: 0.7%

2. The reputation function

The values of increment, r_+, and decrement, r_-, of a neighbor's reputation index as a result of packet delivery events also affect the performance of our mechanism. One way to compare reputation functions is by using the ratio r_- / r_+. The higher the ratio r_- / r_+, the faster the isolation of selfish nodes, but also the higher the number of false positives. Next, we present three heuristics for reputation functions.

i. Double Decrement/Single Increment Ratio (DDSIR)

In this scheme, for each successfully delivered packet, a node increments the reputation index of the next-hop neighbor that forwarded the packet by $r_+ = n$, where n is a positive constant. For each failed delivery a node decrements the reputation index of the neighbor by $r_- = 2n$. This scheme is not very aggressive in penalizing neighbors, but at the same time mandates a node to deliver at least 66% of the packets forwarded through it in order to maintain a fixed reputation.

ii. Hops Away From Source (HAFS)

This scheme is more aggressive than DDSIR in penalizing nodes for dropped packets. In this variation of the algorithm, a node decrements the reputation index of a neighbor as a function of the number of hops h between the node and the source

of the dropped packet: $r_{-} = 2n + m * h$ and $r_{+} = n$, where *n* and *m* are positive constants. The number of hops from the source of the dropped packet could be estimated from the intermediate node's routing table. Thus, on a route where a packet delivery failure event occurs, the reputation index of the node closest to where the event occurred is decremented the most. The motivation of the algorithm is to avoid draining the reputation of nodes along a route that includes a selfish node before isolation takes place. This could occur if nodes along that route are penalized equally by their previous hop neighbors (as in the example discussed in figure 1).

iii. Random Early Probation (REP)

This scheme rewards and penalizes nodes similarly to DDSIR. Additionally, a node randomly rejects participation in a route with neighbors whose reputation is between r_0 and r_{thresh}. As a neighbor's reputation approaches r_{thresh} in a node's table, the probability of the node's participation in a route with this neighbor decreases.

III. Evaluation

Using ns-2, we study the performance of our mechanism. and compare the three proposed reputation functions. We simulate different static ad hoc networks of N^2 nodes arranged as an $N \times N$ grid. The communication range is set such that each node has 4 neighbors, with the exception of edge nodes, which have 3 neighbors, and corner nodes, which have 2 neighbors.

We randomly generate sets of 80, 100, 120, and 140 CBR (Constant Bit Rate) TCP flows, each sending 500Kbits of data. For each set, 25 simulations are executed with flow source and destination pairs selected at random for each run. We rely on TCP acknowledgements and retransmissions as indications of successful and failed packet delivery events, respectively.

Our scheme is routing protocol independent, and we choose to apply it to AODV for the current set of simulations. AODV's specifications allow a node to maintain a blacklist of neighbors [9]. Neighbors identified as selfish (i.e. nodes whose reputation index is below the threshold) are included in this blacklist. All route requests (RREQs) and route replies (RREPs) forwarded by such neighbors will be ignored. Moreover, our protocol allows a node to trigger a RREQ if it detects a route entry in its routing table with a blacklisted next-hop neighbor, thus eliminating all selfish nodes from routes. Eventually, selfish nodes will be identified, eliminated from all routes, and isolated from the network.

1. Comparing the proposed reputation functions

To compare the performance of the three proposed schemes, we use a 4x4 topology with 3 selfish nodes, and an 8x8 topology with 5 selfish nodes, as illustrated in figure 3. For each simulation run, data is collected for selfish nodes' *exposure* (fraction of the neighbors of a selfish node that identified it as such) and *false positives* (fraction of links removed from the network due to a node falsely identifying one of its neighbors as selfish). For all simulation, we set constants *n = 1* and *m = 0.5*.

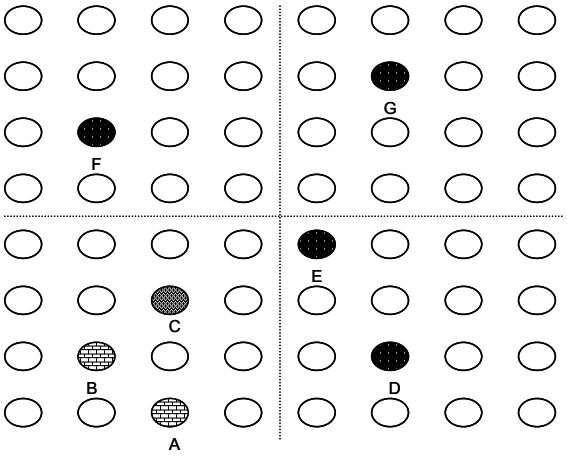

Figure 3. Topologies used in our simulations: 8x8 with 5 selfish nodes (c, d, e, f, and g); the lower left quadrant shows a 4x4 topology with 3 selfish nodes (a, b, and c)

a) Observations

Results for a 4x4 topology shown in figures 4 and 5 demonstrate that the exposure value is consistently lower for REP than for the other two reputation schemes tested. This is expected since REP is the least aggressive scheme among all three. Our results also show that DDSIR and HAFS achieve comparable values of exposure. The results for false positives are ordered based on the aggressiveness of the scheme, with REP achieving the lowest values, followed by DDSIR and then HAFS.

The results for an 8x8 topology are also shown in figures 4 and 5. The exposure results are sorted based

on the aggressiveness of the scheme: REP achieved the lowest exposure, followed by DDSIR and HAFS. For a higher number of flows the exposure of DDSIR approached that of HAFS. On the other hand, the results for false positives exhibit a different trend from those for the 4x4 topology. HAFS was the most aggressive and it had the highest number of false positives. However, the number of false positives generated by REP was either higher or comparable to that of DDSIR. A discussion of these results follows.

b) Analysis

i) Effect of average number of hops on the performance of HAFS

The simulation results for the 4x4 topology indicate that HAFS and DDSIR achieve comparable results in terms of exposure. This is somewhat surprising, since HAFS is a more aggressive scheme than DDSIR. However, looking at the results for the 8x8 topology, HAFS outperforms DDSIR in terms of isolation. These results show the dependency of the performance of HAFS on the average number of hops that a flow traverses. The results also explain why the effect of HAFS can not be seen for networks with a short radius, such as our 4x4 topology (the average number of hops for this topology is 2.67, compared to 5.36 for the 8x8 topology).

It is also apparent from figure 5 that the exposure results for DDSIR approach those of HAFS as the traffic load increases. However, HAFS will reach the same level of exposure faster than DDSIR, as shown in figure 6. This also explains the comparable exposure results of HAFS and DDSIR for the 4x4 topology, since 80 flows is a high traffic load for such a network.

ii) Effect of asymmetric routes on the performance of REP

Simulation results for REP on a 4x4 topology showed the lowest false positives amongst the three algorithms. However, the same results for the 8x8 topology showed unexpectedly high false positives for REP as compared to DDSIR for some of the flow sets simulated. We noticed an increase in the number of asymmetric routes with REP as compared to DDSIR. Since in our protocol symmetric routes lead to a more accurate assessment of neighbors' cooperation, such an increase in the number of asymmetric routes resulted in an increase in the number of false positives.

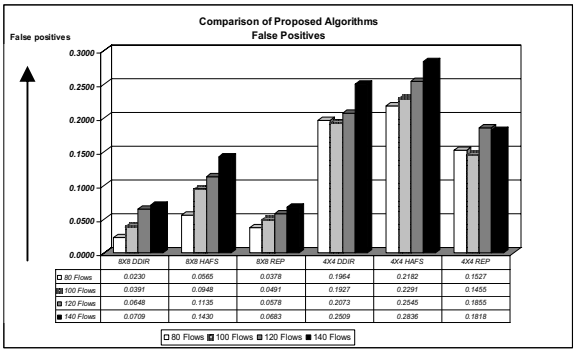

Figure 4. Comparison of false positives for the 3 proposed schemes

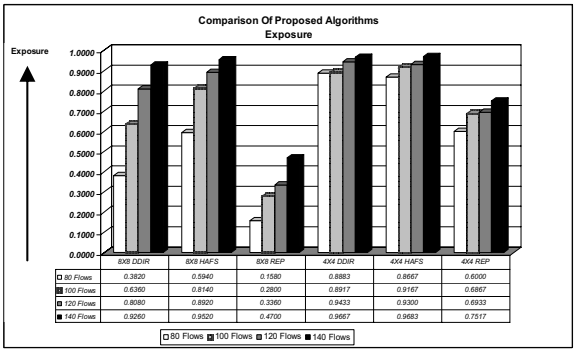

Figure 5. Comparison of exposure for the 3 proposed schemes

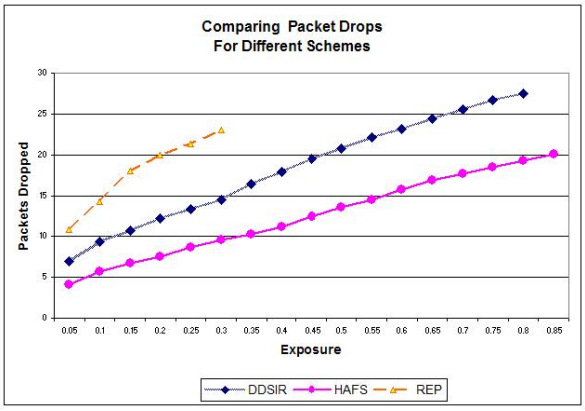

Figure 6. Comparing the 3 schemes based on packets dropped by selfish nodes at each level of exposure. (Results shown are for 120 flows in an 8x8 topology. Similar trend was seen for all other simulations)

Based on these results, we adopted DDSIR for our reputation function, as it consistently generated low false positives and high exposure.

2. Impact on Network Goodput

Using simulations, we calculated the average network goodput in a network with selfish nodes but no reputation mechanism (defenseless network), and another with no selfish nodes and no reputation mechanism in place (safe network). We use the 8x8 topology with 5 selfish nodes simulation setup described previously. Our simulation results presented in figure 7 show, as expected, that DDSIR achieves higher goodput than obtained in a defenseless network. A defenseless network will experience a high number of retransmissions and a high number of flows (around double the number observed in DDSIR) that were unable to deliver any packets to their intended destinations. This is due to the presence of selfish nodes in the network that:

- Drop many packets, resulting in timeouts and high retransmission rates for these flows; and
- Affect the establishment of credible routes to their intended destinations.

Our mechanism was able to avoid such problems by isolating selfish nodes from all routes, resulting in a higher goodput. The presence of selfish nodes decreases goodput by over 11% when no reputation mechanism is employed, while our mechanism limits that decrease to less than 5.5%.

3. Effect of Mobility

In order to asses the effect of node mobility on the performance of the proposed scheme, we ran a number of ns-2 simulations using the random way point mobility model. We used three sets of simulations of 64 nodes with varying average node speed and pause time. For each simulation, we calculated the exposure of selfish nodes and the number of packets dropped by each selfish node for each of its neighbors. Our results show that the average number of packets dropped by a selfish node per neighbor decreased as the average node speed increases (figure 8), which led to slower isolation (table 2). This is expected due to the decrease of the average interaction time between nodes as their speed increases. As a result, a node will forward fewer packets through a single neighbor. Thus, the impact of the selfish behavior of a node is minimized. This leads to the conclusion that, for mobile nodes, isolation is inversely proportional to speed and mobility tends to reduce the impact of selfishness on the network.

Table 2.
Effect of average node speed and pause time on isolation of selfish nodes.
Each value represents percentage of selfish nodes isolated averaged over 10, 600-sec. simulations.

Pause/ Avg.Speed	0.1 meter/s	5 meter/s	15 meter/s
Pause 0	**24%**	**1%**	**0.35%**
Pause 1	**28%**	**0.97%**	**0.21%**
Pause 2	**29%**	**0.5%**	**0.42%**

4. Improvements to Current Mechanism

a) Storage overhead

The fact that nodes have to store packet traces in order to distinguish between delivered and retransmitted packets results in storage overhead. We ran a number of simulations in order to observe the impact of limiting the size of the lookup table on the overall protocol performance (isolation time). We tested against two heuristics:

- Use a FIFO lookup table of 1000 entries; and
- Assign an expiration time per table entry, which is based on estimated flow round-trip time (RTT) at each node. Upon timing out, the entry is eliminated from the lookup table.

Our results indicate that there is a tradeoff between the lookup table size and the isolation time. By setting an upper limit on the lookup table size, there was a slight increase in the isolation time.

As stated earlier, the size of the lookup table could be further minimized by using hashing algorithms. For 1000 table entries, the size of the table is about 20Kbytes. By using a hash value of 2bytes ($1/2^{16}$ collision probability), the table size could be further reduced to 2Kbytes.

b) Reliance on TCP

As the protocol relies on acknowledgement produced by the destination, there is the question of its applicability to flows employing UDP. We believe that we can rely on some application layer end-to-end acknowledgement mechanism for such scenarios, such as provided by the Real-Time Streaming Protocol (RTSP).

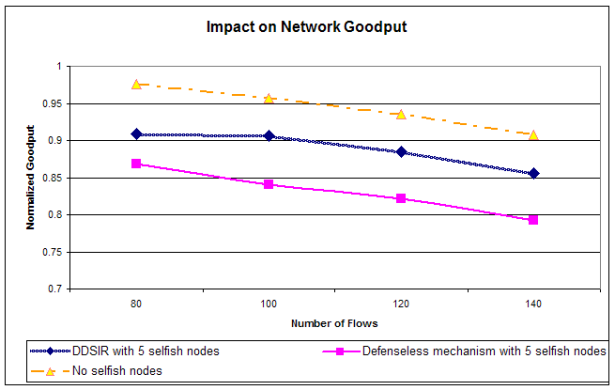

Figure 7. Impact of the use DDSIR on the network goodput

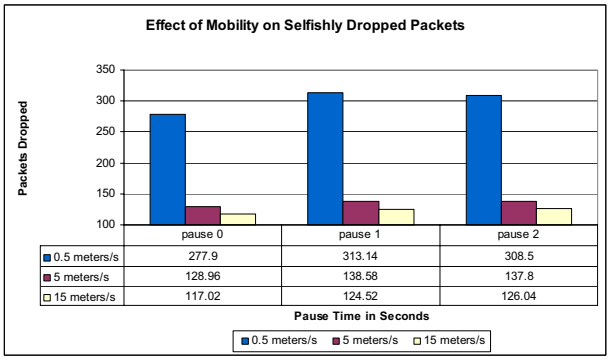

Figure 8. Effect of average node speed and pause time on packets dropped by selfish nodes

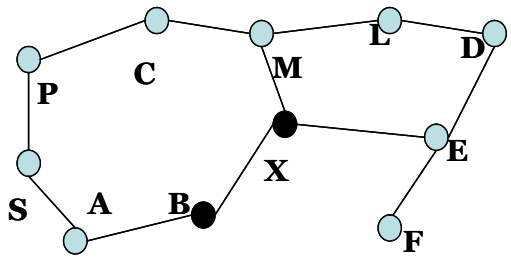

Figure 9. An ad hoc-network with colluding nodes (B and X)

5. Possible Attacks

Since our mechanism was designed to handle primarily selfish behavior, it is susceptible to a number of malicious attacks. We plan to investigate possible extensions to this work, including mechanisms for providing authentication, confidentiality, and message integrity in order to deter, detect, and isolate malicious behavior.

One of the advantages of our mechanism, as compared to other mechanisms, is its robustness to collusion. Schemes that rely on sharing reputation information require intermediate nodes to inform other nodes in the network about a neighbor's selfish/malicious behavior. Consider the simple ad hoc network shown in figure 9. If nodes B and X are in collusion, node B could either not broadcast information regarding bad behavior by X (a problem in schemes that only maintain negative reputation values), or falsely report good behavior by X (a problem in schemes that increase reputation based on positive reporting). Since our mechanism does not involve exchange of reputation information, it is robust against colluding nodes.

IV. Related Work

The use of a reputation scheme to judge a node's intent is one of the techniques adopted to detect and isolate selfish nodes in an ad hoc network. Different reputation mechanisms appear in the literature. We can broadly classify these into two categories:

1. Mechanisms in which nodes exchange reputation among themselves; and
2. Mechanisms in which nodes independently assess their neighbors' reputation based on direct interactions.

A large number of schemes [1] [2] [3] [4] [5] [7] [8] belong to the first category, with varying implementations. One advantage of such schemes could be their quick convergence in detecting node misbehavior, especially in a large ad hoc network, due to increased information regarding a particular node's behavior. However, this approach has two potential drawbacks: they often assume that nodes that send reputation information about their peers are themselves trustworthy; and they are subject to collusion among nodes that misreport reputation information. The algorithm proposed in [6] belongs to the second category and deals with establishing reputation only via direct interactions with other nodes. This is similar in concept to our proposed mechanism, but the analysis and inferences reached are limited as compared to the work reported here.

In schemes based on exchange of reputation information, the reputation index a node assigns to others in the network is based on a combination of directly observed behavior (direct interaction) and reported behavior (indirect interactions). In order to overcome the problem of trust in the reporting of reputation information, [1] and [5] propose an approach that maintains a reputation value for every function in the ad hoc network, including the

reporting function itself. The information obtained from the various reporting nodes is then individually weighted based on the reputation of the reporting node. This solution addresses the problem of trusting indirectly observed behavior, but introduces complexity in maintaining different reputation values for each network functionality. It also does not fully address the problem of collusion. Two malicious nodes, say A and B, can mount a denial of service attack where A systematically reports positive reputation information about B, ensuring that B remains in the routing tables for other nodes in the network, while B systematically drops any packets routed through it. Such collusion scenarios are difficult to prevent unless the reputation mechanism relies primarily on assessing reputation based on direct interaction with other nodes, as is the case of the mechanism reported in this paper.

Both [2] and [4] rely on nodes' operating in the promiscuous mode in order to assess whether their neighbors are correctly forwarding packets. However, it is difficult to constantly switch the network interface between the transmit/receive and promiscuous modes. In addition, the wireless nature of the medium makes the method error prone. Our reputation scheme relies on feedback from the destination to assess node behavior and, therefore, the interfaces do not have to be switched to a promiscuous mode of operation.

Our reputation mechanism, though developed independently, is similar in some respects to the work reported in [6]. In [6], only a single reputation function (simple increment/decrement by a constant) is used, and its effectiveness in the fast isolation of selfish nodes or the reduction of false positives is not reported. Our mechanism uses more sophisticated reputation functions and we provide a detailed comparative analysis of three heuristics for the selection of an appropriate reputation function. We demonstrate that the reputation function has a major impact on the node isolation time and the percentage of false positives. A focus of our current research is the optimization of the reputation function. Another major difference is the impact of mobility on isolating selfish nodes. We evaluated our work under different mobility scenarios and observed using simulation that a highly mobile environment leads to a higher isolation time.

V. Conclusions and Future Work

In this paper, we proposed and described the design and evaluation of a reputation-based mechanism that isolates selfish nodes in an ad hoc network. Our results indicate that the mechanism is successful in achieving fast isolation of selfish nodes while maintaining false positives at a reasonably low level.

We are currently investigating extensions to the protocol to detect and isolate various forms of malicious behavior emphasizing autonomous decisions by individual nodes. We are also interested in investigating the effect of congestion on the protocol's performance. Additionally, we will explore reputation rebuilding mechanisms, which allow a node that was labeled selfish and isolated from the network to be re-evaluated and reinstituted into the network.

VI. References

[1] P. Michiardi and R. Molva, "CORE: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," Proceedings of the 6th Joint Working Conference on Communications and Multimedia Security, September 2002, pp. 107-121.

[2] S. Buchegger and J.Y. LeBoudec, "Performance analysis of the CONFIDANT protocol: cooperation of nodes – fairness in dynamic ad hoc networks," Proceedings of the ACM MobiHoc, June 2002.

[3] P. Dewan and P. Dasgupta, "Trusting routers and relays in ad hoc networks," Proceedings of the International Conference on Parallel Processing Workshops, October 2003, pp. 351-359.

[4] S. Marti et al., "Mitigating routing misbehavior in mobile ad hoc networks," Proceedings of Sixth Annual IEEE/ACM Intl. Conference on Mobile Computing and Networking, April 2000, pp. 255-265.

[5] J. Liu and V. Issarny, "Enhanced reputation mechanism for mobile ad hoc networks," Proceedings of the 2nd Intl. Conference on Trust Management, April 2004.

[6] P. Dewan, P. Dasgupta and A. Bhattacharya, "On using reputations in ad hoc networks to counter malicious nodes," Proceedings of the 10th Intl. Conference on Parallel and Distributed Systems, July 2004, pp. 665-672.

[7] S. Buchegger and J.Y. LeBoudec, "The effect of rumor spreading in reputation systems for mobile ad hoc networks," Proceedings of the Workshop on Modeling and Optimization in Mobile, Ad hoc and Wireless Networks, March 2003.

[8] Z. Despotovic and K. Aberer, "A probabilistic approach to predict peers' performance in P2P networks," Proceedings of the Intl. Workshop on Cooperative Information Agents, September 2004.

[9] C. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-demand Distance Vector (AODV) routing," IETF RFC 3561, July 2003; www.faqs.org/rfcs/rfc3561.html.

[10] P. Michiardi and R. Molva, " Simulation based analysis of security exposures in mobile ad hoc networks," Proceedings of the European Wireless Conference, February 2002.