

# Distributed Detection of Selfish Routing in Wireless Mesh Networks

Bo Wang    Sohraab Soltani    Jonathan K. Shapiro    Pang-Ning Tan    Matt Mutka  
Michigan State University

**Abstract**—Community wireless mesh networks are vulnerable to free riders who refuse to forward data for others. As the naive selfish strategy of dropping data is readily detected, free-riders are motivated to subtly manipulate the routing protocols to minimize their workload while still evading detection. This paper presents an adaptive on-line algorithm to detect such selfish behavior based solely on local observations of messages exchanged by AODV-like routing protocols. We use a finite state machine model of locally observable protocol actions to generate a statistical description of the behavior of each neighbor and apply statistical analysis to cluster neighboring nodes on the basis of behavioral similarities and identify the selfish ones. Through simulation, we evaluate the performance of our method with respect to the probability of detecting selfish nodes and the rate of false positives against two generic selfish strategies—dropping route requests and dropping route replies. We also evaluate the effect of detection on an adaptive adversary who attempts to operate as selfishly as possible while still evading detection. We find that our technique can detect dropped route requests as well or better than a variant of the widely-used Watchdog detection method, with a lower rate of false positives. In the case of dropped route replies, a fundamental scarcity of observable routing events prevents any algorithm from performing well, suggesting the need to revisit the design of routing protocols.

## I. INTRODUCTION

Community mesh networks promise to provide a last-mile wireless access network with bandwidth comparable to that of a centrally managed wireless ISP in places where no robust provider infrastructure exists [1], [2]. Access is provided by a self-organizing network of wireless routers controlled by end-users, which forward data on behalf of others. As with any end-user supported infrastructure, ubiquitous cooperative behavior cannot be assumed a priori. Scarce access bandwidth and power, as well as security concerns, may induce some users to avoid forwarding data for other nodes, even as they send their own traffic through the system. Such free-riding behavior, if widespread, threatens the performance of community networks.

The impact of non-cooperative behavior on the capacity of ad hoc wireless networks has previously been

observed [3] and several general techniques have been adapted to such networks to reduce the incentive for free-riding. Of particular interest in this work is the class of techniques collectively known as *reputation mechanisms*, which aim to identify bad actors in a system and then isolate or punish them. A reputation mechanism is typically composed of several distinct parts: (a) a method of evaluating the behavior of others based on direct observations, (b) an optional component to improve convergence by exchanging locally gathered reputation information with (possibly untrusted) nodes, and (c) some punishment mechanism to provide the incentive for cooperation. Our work focuses on the first of these components—the local, or first-hand, assessment of reputation—which forms the foundation of any reputation mechanism. Without reliable local reputation assessment, additional exchange of reputation information may not improve detection accuracy, even if information is only exchanged with trusted nodes. Furthermore the threat of punishment may be ineffective as an incentive if selfish actors are not reliably detected or if cooperative ones are falsely identified as selfish [4].

Several reputation mechanisms have been proposed for encouraging cooperation in peer-to-peer systems including some specifically designed for ad hoc networks [5], [6], [3]. Early work on detecting selfish behavior in ad hoc networks focused on observing nodes' failure to forward data. The basic mechanism for detecting such failures was first proposed by Marti, et al, who dubbed the technique *Watchdog* [7]. Watchdog uses a so-called *implicit acknowledgement* obtained by overhearing a retransmitted packet. By promiscuously listening for an implicit acknowledgement, a node can ascertain with reasonable certainty that the its successor on a path has forwarded a packet. Watchdog is quite successful at detecting dropped data—so successful that a selfish user wishing to evade detection would be well advised to forward data when asked but take actions to minimize the chances of being selected for forwarding in the first place. More sophisticated selfish behavior, therefore, is likely to involve manipulation of routing information.

In this work, we consider the problem of detecting selfish routing behavior based purely on local observa-

tions of an on-demand ad hoc routing protocol such as AODV [8].<sup>1</sup> We employ a specification-based approach, which exploits the structure of such protocols to extract useful features from the observed transmissions of neighboring nodes. We then apply statistical inference and clustering techniques to the data to classify neighboring nodes as either selfish or cooperative.

Our technique requires no training data but instead compares observed behavior of multiple neighbors against each other, providing a basis for a fully distributed adaptive on-line local reputation assessment algorithm. Because it requires data collection over timescales on the order of minutes, this algorithm is best suited for networks with little or no mobility of forwarding nodes such as community mesh networks.

The remainder of this paper is organized as follows: In Section II we review the operation of the AODV routing protocol and develop a finite state machine representation of locally observable node behavior. We present our detection algorithms in Section III. Section IV presents simulations of two types of selfish attack and the detection of both naive and strategic selfish adversaries. In Section V reviews related work and we conclude in Section VI with a discussion of potential directions for further study.

## II. SPECIFICATION-BASED APPROACH

Consider an ad hoc network in which routing is accomplished by a protocol with the basic structure of AODV. The AODV routing protocols works by propagating a route request (RREQ) message throughout the entire network by means of a flooding broadcast. Route reply (RREP) messages are then unicast over a subset of the reverse broadcast paths, providing the request originator with one or more candidate routes. Nodes may cache the routes they discover in either the flooding or response phases and respond to route requests with cached information.

Selfish nodes in this network attempt to manipulate the routing protocol to minimize their chances of being included on routes for which they are neither source nor destination. In processing routing messages, a selfish node can choose among several actions to accomplish this goal. The most effective action dropping or otherwise tampering with broadcast RREQ messages, which ensures that no routes will ever be selected through the selfish node. An alternative approach is to drop, delay, or modify RREP messages, which prevents replies originating downstream from reaching

<sup>1</sup>Although we will frame our discussion in terms of AODV specifically, the techniques we present apply generally to any on-demand protocol which uses a flooding route discovery phase followed by a reverse-path route identification phase.

State	Description
1:init	No RREQ observed
2:unexp RREP	Observed receipt of an "unexpected" RREP
3:rcvd RREQ	Observed receipt of RREQ
4: fwd RREQ	Observed broadcast of RREQ
5:timeout RREQ	No activity observed after receipt of RREQ
6:rcvd RREP	Observed receipt of RREP
7:complete LRI	Observed forwarding of valid RREP
8: timeout RREP	No activity observed after receipt of RREP

TABLE I: Interpretations of the states in Fig. 1.

the RREQ source. Dropping RREQs cannot, however, prevent upstream nodes from responding with previously cached reverse route information.

Consider the perspective of an individual node—referred to hereafter as the local node—sharing wireless links with some number of neighboring nodes—who wishes to determine which, if any, of its neighbors is behaving selfishly based solely on local observations. In reasoning about possible solutions for this problem, it is helpful to characterize the data from which these observations can be constructed.

We refer to the set of transmissions that flood a single RREQ message throughout the network along with the set of RREP messages transmitted in response as a *routing instance*. Although the local node cannot observe the entire set of transmissions for any particular routing instance, it can observe a subset, which we refer to as a *local routing instance* (LRI). An LRI includes transmissions sent by the local node, itself, as well as those sent by its neighbors, including messages overheard by the local node but not explicitly addressed to it. The set of all observed LRIs comprises the data that the local node can examine for evidence of selfish routing behavior.

### A. Finite State Machine Model

To impose additional structure on the observations, the local node associates each transmission in a LRI with its sender and its receiver, or, in the case of its own outgoing broadcast RREQs, with multiple receivers.<sup>2</sup> The possible sequences of transmissions in a LRI are determined by the AODV protocol. We describe this set of possible behaviors using the finite state machine (FSM) description shown in Fig. 1. Table I gives the meanings of each state in the FSM.

The FSM in Fig. 1 describes the behavior of one particular neighbor with respect to a single LRI. Each transmission observed by the local node is recorded as a state transition in one or more neighbors' FSMs. It is worth emphasizing that the FSM does not model the

<sup>2</sup>In the case of RREQs broadcast by the local node itself, we assume such transmissions are received by all current neighbors, despite the possible absence of link-layer acknowledgements for broadcast transmissions.

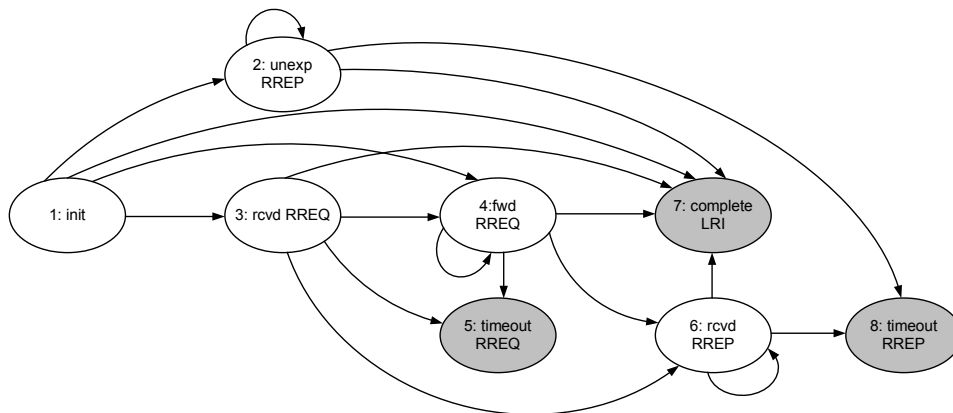


Fig. 1: Finite state machine model of a monitored node with respect to a single LRI, as observed by its neighboring local node. Final states are shaded.

exact behavior of a node participating in AODV routing. Instead, it models the node’s behavior *as observed by its neighbor, the local node*. For clarity in the following discussion, we focus on one particular neighbor being observed by the local node as the *monitored node*. It should be understood, however, that the local node simultaneously monitors all its neighbors

The routing messages observed by local node over the course of its lifetime in the system constitute a series of interleaved LRIs. Each LRI can be identified by the combination of source and destination contained in a RREQ message. We denote the identifier for the  $k^{\text{th}}$  LRI observed by a node as  $(s_k, d_k)$ . Note that the combination  $(s_k, d_k)$  does not uniquely identify a LRI since a source can issue multiple RREQs for the same destination. However, we find that this ambiguity has little effect on our analysis as long as subsequent re-issued requests are suitably delayed so that there is only one active LRI for a given  $(s_k, d_k)$  at any point in time. In practice, AODV sources explicitly try to respect such delays.

Prior to participating in an LRI—that is, prior to being observed to have received a RREQ for a particular  $(s_k, d_k)$ —the monitored node begins in the initial state 1. As the local node observes the monitored node’s behavior over the course of the LRI, it records a sequence of transitions from this initial state to one of three possible final states.

If the local node broadcasts a RREQ, it assumes that the monitored node receives it and records the transition  $1 \rightarrow 3$  for that neighbor’s FSM. If the monitored node is observed to broadcast a RREQ, either the  $1 \rightarrow 4$  or  $3 \rightarrow 4$  transition is recorded, depending on whether the local node previously broadcast the same RREQ. Transitions to timeout states occur when the local node fails to observe any additional activity for the LRI within a suitable duration. Transitions to final state 7 (complete LRI) occur when the monitored node is observed to

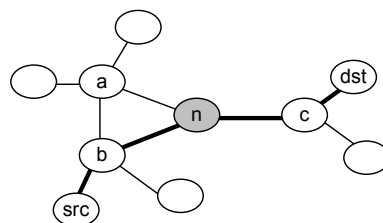


Fig. 2: Example of the local routing instance observed by node  $n$  for the establishment of the route indicated with bold lines. For this example, all nodes are assumed to behave cooperatively.

forward a RREP.<sup>3</sup> It is in this state that the monitored node becomes a candidate for inclusion on a route.

As an example, consider the LRI observed by a local node  $n$  during the discovery of a route from the source to the destination shown in Fig. 2. Table II shows the events observed by  $n$  and the corresponding state transitions in FSMs representing each of its three neighbors.

Upon reaching a final state, the FSM is considered complete and the corresponding sequence of transitions can be stored by the local node and associated with the monitored node. After accumulating a sufficient number of such sequences for all of its neighbors, the local node performs a statistical analysis on the data, which we describe in the following section.

### III. DETECTING SELFISH BEHAVIOR

Recall that the local node records a completed sequence of transitions observed for a monitored node in each LRI. The completed sequences accumulated

<sup>3</sup>In this discussion, we have implicitly assumed that selfish nodes either drop messages or forward them without maliciously modifying message fields. To the extent that the local node can observe the modification of routing messages, the FSM can be readily extended to account for invalid forwarding actions, such as forwarding a RREP with an inflated hop count.

neighbor	event	state transition
a	a broadcasts RREQ	1→4
	n broadcasts RREQ	4→4
	timeout	4→5
b	b broadcasts RREQ	1→4
	n broadcasts RREQ	4→4
	n sends RREP to b	4→6
	b sends RREP to src (overheard)	6→7
c	n broadcasts RREQ	1→3
	c broadcasts RREQ	3→4
	c sends RREP to n	4→7

TABLE II: Assignment of observed transmission events to neighbors and the corresponding state transition sequences for the three neighbors of node  $n$  in Figure 2

over time can be efficiently summarized in a transition matrix  $T = [T_{ij}]$  containing the total number of times each transition  $i \rightarrow j$  has been observed. The local node runs an online detection algorithm every  $W$  seconds using data from the most recent  $D = dW$  seconds of observations, where  $d$  is a small integer. We refer to  $D$  as the *detection window* of the algorithm. The value of  $D$  should be small enough to allow responsive punishment, but large enough to accumulate sufficient data to support accurate detection.

#### A. Detection algorithm overview

Although a transition matrix summarizes the local routing behavior of a monitored node, determining the selfish behavior of a node based on its local transition probabilities alone may not be sufficient. This is because selfishness is a relative term, with respect to the behavior of other monitored nodes. By comparing the transition matrices of a collection of nodes, one might be able to detect selfish nodes with higher confidence than looking at the transition matrix of each node individually. This motivates the approach used by our detection algorithm, which initially clusters the neighbors of the local node and then classifies the clusters into selfish or cooperative nodes. In designing the algorithm, we are faced with several challenges.

First, the clustering output is sensitive to the presence of noise in the data. To address this problem, we develop a statistical-based approach for clustering the monitored nodes based on pairwise comparisons of their transition matrices. Instead of using standard measures for clustering (such as Euclidean distance), we develop a more robust measure called *conditional dissimilarity*.

Second, while clustering can identify sets of neighbors with statistically similar behavior, it does not determine which clusters appear to contain selfish nodes. To perform this classification, we introduce an additional measure of cooperation, which we refer to the *cooperation score*, described below. The cooperation score

for a cluster of neighbors is taken to be the mean score over the entire set.

Finally, it is important to recognize that cooperation is only one of several influences on observed node behavior; other factors such as the degree of connectivity or location within the topology may interact with selfishness to produce complex behavior classes. The implications of this complexity are twofold. We must select the granularity of clustering without any a priori idea notion of what would be appropriate. Second, because many factors influence behavior, there is no guarantee the clusters we discover will accurately separate the neighbors *with respect to their cooperation scores*. Since all nodes in a cluster share the same classification, poorly chosen clusters may lead to many false classifications. We also, therefore, develop a test based on the Analysis of Variance (ANOVA) among clusters to determine whether clustering is informative for the purposes of classification.

We address the challenges outlined above with an iterative algorithm (Algorithm 1) that searches for the smallest set of clusters that accurately separate neighbors by cooperation score. We favor fewer (larger) clusters because confidence in the mean cooperation score improves with cluster size. The algorithm is conservative, preferring to allow some selfish nodes to remain unidentified rather than falsely accusing cooperative nodes. Thus, if two or more clusters are identified by the algorithm, only nodes in the cluster with the lowest average cooperation score are classified as selfish. Moreover, if at any point the algorithm determines that clustering is not informative, all nodes are classified as cooperative.

In the rest of this section, we describe various components of Algorithm 1 in greater detail. In the following discussion, we take the perspective of a single node monitoring its neighboring nodes, indexed  $1, \dots, R$ . Let  $T^{(r)} = [n_{ij}^{(r)}]$ , denote the observed transition matrix for the  $r^{th}$  neighbor, where  $n_{ij}^{(r)}$  is the number of transitions from state  $i$  to  $j$  observed in the preceding detection interval. Let  $m$  denote the number of states in the FSM, making each  $T^{(r)}$  size  $m \times m$ .

#### B. Measuring neighbor similarity

Let  $T_i^{(r)} = (n_{i1}^{(r)}, \dots, n_{im}^{(r)})$  denote the  $i^{th}$  row of the transition matrix  $T^{(r)}$ , describing transitions out of state  $i$  at neighbor  $r$ . Since behavior in state  $i$  is independent of other states, the distribution over possible transitions out of state  $i$  satisfies a multinomial distribution. If transitions from state  $i$  are identically distributed for

---

**Algorithm 1** Selfish Neighbor Discovery Algorithm
 

---

**Input:**  $T^{(1)}, \dots, T^{(R)}$  /\*transition matrices\*/  
**Input:**  $\alpha$  /\*Pearson confidence parameter\*/  
**Input:**  $\beta$  /\*ANOVA confidence parameter\*/  
**Returns:**  $\mathcal{S}$  /\*set of selfish neighbors\*/  
**for all** neighbor pairs  $(r, s)$  **do**  
 $L_{rs} = \text{PearsonSimilarity}(T^{(r)}, T^{(s)}, \alpha)$   
**end for**  
 /\*Notation:  $L = [L_{rs}]$  (pairwise similarities)\*/  
**for all** neighbor pairs  $(r, s)$  **do**  
 $a_{rs} = \text{ConditionalSimilarity}(L, r, s)$   
**end for**  
 /\*Notation:  $A = [a_{rs}]$  (conditional dissimilarities)\*/  
 $\pi_1 = 1$   
**for**  $k = 2$  to  $R$  **do**  
 $\mathcal{C}_k = \text{Cluster}(A, k)$  /\*generate  $k$  clusters\*/  
 $\pi_k = \text{ANOVATest}(\mathcal{C}_k)$  /\*test for significance\*/  
**if**  $\pi_k < \beta$  **then**  
 $\mathcal{S} = \arg \min_{c \in \mathcal{C}_k} \text{MeanCooperationScore}(c)$   
 return  $\mathcal{S}$  /\*return least cooperative cluster\*/  
**else if**  $\pi_k > \pi_{k-1}$  **then**  
 return  $\emptyset$  /\*no monotonic improvement\*/  
**end if**  
**end for**  
 /\*end here if  $\pi_R \geq \beta$ \*/  
 return  $\emptyset$

---

two neighbors  $r$  and  $s$ , we write  $T_i^{(r)} \stackrel{d}{=} T_i^{(s)}$ .<sup>4</sup>

We test the hypothesis  $T_i^{(r)} \stackrel{d}{=} T_i^{(s)}$  using the Pearson statistic [9], which is defined as

$$Q_{2(m-1)}^{(rs)}(i) = \sum_{l \in \{r, s\}} \sum_{j=1}^m \left[ n_{ij}^{(l)} - \bar{n}_{ij}^{(l)} \right]^2 / \bar{n}_{ij}^{(l)}, \quad (1)$$

$$\bar{n}_{ij}^{(l)} = N_i^{(l)} \frac{n_{ij}^{(r)} + n_{ij}^{(s)}}{N_i^{(r)} + N_i^{(s)}},$$

where  $N_i^{(r)}$  and  $N_i^{(s)}$  respectively denote the total number of transitions from state  $i$  in  $T^{(r)}$  and  $T^{(s)}$ .

The statistic  $Q_{2(m-1)}^{(rs)}(i)$ , whenever it is well defined<sup>5</sup>, has an asymptotically chi-square distribution with  $m-1$  degrees of freedom,  $i = 1, \dots, m$ . We reject the hypothesis  $T_i^{(r)} \stackrel{d}{=} T_i^{(s)}$  at the level of significance  $\alpha$  if

$$Q_{2(m-1)}^{(rs)}(i) > \chi_{m-1, \alpha}^2.$$

<sup>4</sup>We will later abuse this notation by writing  $T^{(r)} \stackrel{d}{=} T^{(s)}$ , if  $T_i^{(r)} \stackrel{d}{=} T_i^{(s)} \forall i \in \{1, \dots, m\}$ .

<sup>5</sup>The quantity  $Q_{2(m-1)}^{(rs)}(i)$  is well defined if transition frequencies  $n_{ij}^{(r)} \geq 1$ ,  $n_{ij}^{(s)} \geq 1$ . In practice this condition can be enforced by simply increasing all the observed transition frequencies by one unit. For sufficiently large observation windows, this adjustment has a negligible effect on the transition probabilities.

For notational convenience we define the indicator

$$B_i^{(rs)} = \mathbb{1}[Q_{2(m-1)}^{(rs)}(i) > \chi_{m-1, \alpha}^2].$$

Thus, we obtain an approximation for the probability of falsely rejecting the Pearson hypothesis,

$$P(B_i^{(rs)} = 1 | T_i^{(r)} \stackrel{d}{=} T_i^{(s)}) \approx \alpha. \quad (2)$$

We consider the inverse conditional probability  $P(T_i^{(r)} \stackrel{d}{=} T_i^{(s)} | B_i^{(rs)})$  to be a reasonable measure of the similarity of  $r$  and  $s$  with respect to state  $i$ . In the absence of prior information, we make neutral assumptions, setting the unconditional probabilities that  $r$  and  $s$  differ in state  $i$ , and that the Pearson test rejects its hypothesis to 0.5. Under these assumptions, Bayes' formula yields

$$P(T_i^{(r)} \stackrel{d}{=} T_i^{(s)} | B_i^{(rs)=1}) \approx \alpha$$

To measure the similarity of  $r$  and  $s$  across all states of the FSM, we evaluate (1) for all rows of  $T^{(r)}$  and  $T^{(s)}$ , obtaining a vector  $B^{(rs)} = [B_i^{(rs)}], \{i = 1, \dots, m\}$ . From the standard Markovian assumption—justified by our FSM model—we may infer that

$$\begin{aligned} L_{rs} &= P(T^{(r)} \stackrel{d}{=} T^{(s)} | B^{(rs)}) \\ &= \alpha^{S^{(rs)}} (1 - \alpha)^{m - S^{(rs)}} \\ &\approx \alpha^{S^{(rs)}}, \end{aligned} \quad (3)$$

where

$$S^{(rs)} = \sum_{i=1}^m B_i^{(rs)}. \quad (4)$$

The approximation (3) is obtained by dropping lower-order terms, which is justified since  $\alpha \ll 1$ .

Observe that for small values of  $\alpha$ ,  $L_{rs} \in [0, 1]$  is strictly decreasing in  $S^{(rs)}$ , the number of rejections of Pearson's hypothesis. It's compliment  $1 - L_{rs}$  can therefore be considered a scalar measure of the *dissimilarity* between neighbors  $r$  and  $s$ <sup>6</sup>, and would, in principle, provide a suitable input for clustering. Due to noise in the data, however, we often find in practice that two similar nodes  $r$  and  $s$ , for which  $L_{rs} \approx 1$ , are inconsistent with respect to a third node  $t$  such that  $L_{rt} \not\approx L_{st}$ . This inconsistency diminishes the effectiveness of clustering.

To improve our ability to reliably cluster the data, we derive a new measure from the pairwise similarities that emphasizes consistency. Inspired by a similar

<sup>6</sup>Informally,  $1 - L_{rs}$  can be interpreted as the distance between  $r$  and  $s$  in an abstract space. Technically, however, the term *distance measure* only applies if the points defined by the distances can be embedded in a metric space. Pairwise similarities generated from noisy data generally do not satisfy this criterion; thus,  $1 - L_{rs}$  is properly referred to as a *dissimilarity*.

measure used in anomaly detection [10], we define the *conditional dissimilarity* of  $r$  and  $s$ , denoted  $a_{rs}$ , as

$$a_{rs} = 1 - \frac{f_{rs}^2}{f_{r/s} * f_{s/r}} \quad (5)$$

where

$$\begin{aligned} f_{rs} &= \sum_{t \neq r, s} \min(L_{rt}, L_{st}) \\ f_{r/s} &= \sum_{t \neq r, s}^K L_{rt} \\ f_{s/r} &= \sum_{t \neq r, s}^K L_{st} \end{aligned}$$

Note that  $a_{rs}$  does not depend on  $L_{rs}$ , the pairwise similarity of nodes  $r$  and  $s$ . Instead,  $a_{rs}$  is a measure of inconsistency in  $r$  and  $s$ 's similarities with all other neighbors. In one extreme case, for example, if  $L_{rt} = L_{st} \forall t \neq r, s$ , then  $r$  and  $s$  are completely consistent and  $a_{rs} = 0$ . Conversely,  $a_{rs} \approx 1$  indicates a high degree of inconsistency. Using the conditional dissimilarity  $a_{rs}$ , we can accurately divide neighbors into behavior classes using any of several standard clustering algorithms; we use single linkage clustering [11].

### C. Cooperation score

To define a cooperation score, we must incorporate additional domain-specific knowledge beyond the FSM protocol specification. We do so by partitioning the transitions in the AODV FSM into a 'good' set  $G$ , a 'bad' set  $B$  and a remaining neutral set. This division reflects the fact that, although no transitions in the FSM are inherently selfish<sup>7</sup>, some transitions ( $G$ ) are more likely to be observed for cooperative nodes whereas others ( $B$ ) are more likely for selfish nodes. The cooperation score for node  $r$  is given by

$$c_r = \frac{\sum_{i,j \in G}^m n_{ij}^{(r)}}{|G|} - \frac{\sum_{i,j \in B}^m n_{ij}^{(r)}}{|B|},$$

A low score indicates a strong suspicion of selfishness.

### D. A test for informative clustering

We use the well-known ANOVA test to evaluate a stopping criterion for Algorithm 1 based on the statistical significance of the differences among the mean cooperation scores of the clusters. ANOVA computes a

<sup>7</sup>Timeout transitions will occur, for example, whenever RREP messages fail to follow the reverse broadcast path through the monitored node or due to channel errors during broadcasts.

measure  $\pi_k$  of the probability that the variation among the mean cooperation scores of  $k$  clusters has simply occurred by chance. A lower value of  $\pi_k$  suggests that the clusters actually represent distinct differences in cooperative behavior. At each iteration of the final loop in Algorithm 1, we generate  $k$  clusters and compare  $\pi_k$  against a chosen level of significance  $\beta$ . If  $\pi_k < \beta$ , we conclude that the clusters accurately separate selfish nodes from cooperative ones and proceed with classification. To be conservative, only nodes in the cluster with lowest mean cooperation score are classified as selfish. If  $\pi_k > \pi_{k-1}$  we interpret the failure to make monotonic progress toward the threshold  $\beta$  as an indication that neighbor similarities are not well explained by their levels of cooperation and classify all nodes as cooperative. Otherwise, we regroup the nodes into  $k+1$  clusters run ANOVA again. If the algorithm reaches the extreme case of generating  $R$  clusters—one for each neighbor—but  $\pi_R \geq \beta$ , we again conclude that clustering is uninformative and classify all neighbors as cooperative.

Note that the ANOVA confidence parameter  $\beta$  can be tuned to adjust the aggressiveness with which selfishness is detected, providing a means to trade detection rate against false positive rate.

## IV. EVALUATION

In this section, we present simulation results for our detection methods in the presence of a varying amount of selfishness in the overall population. Our goals in conducting simulations are three-fold: First, we wish to verify a correlation between the probability that the node drops routing messages and the amount of data forwarded by a selfish node. As we will see, selfish nodes can exploit such a correlation by dropping at a rate low enough to make detection challenging, while still reducing the expected amount of data to be forwarded. Second, we wish to evaluate FSM-based detection while varying the proportion of selfish nodes within the population and the aggressiveness with which those nodes drop messages. Third, we wish to evaluate the robustness of our detection technique against a strategic adversary who explicitly seeks to evade detection. Also we compare our detection technique against a Watchdog-like technique applied to routing traffic.

### A. Simulation setup

Using the NS-2 simulator [12] (version 2.27), we simulate a flat area of 900m by 900m with 50 randomly positioned stationary wireless nodes using AODV for routing. All nodes use a 11Mbps 802.11b radio with the standard NS shadowing propagation model [12]. To

make our simulation more realistic, the configuration of the shadowing model, antennas and wireless cards are based on a measurement study of a real wireless mesh network [13]. A short timeout value of 0.5 seconds triggers FSM transitions based on failure to observe an implicit acknowledgement—i.e. a forwarded or re-broadcast message. A longer timeout value of 3 seconds is used for transitions caused by failing to receive a RREP.

Our simulated nodes perform online selfishness detection as described in Section III, every  $W$  seconds based on the data collected in the previous  $D$  seconds. In our simulation, we use an observation window size of  $W = 100$  seconds. Settings for detection window size  $D$  are discussed below in the presentation of simulation results. The Pearson confidence parameter  $\alpha$  was set to 0.1 for all experiments presented. The ANOVA parameter  $\beta$  was varied as discussed below.

Traffic is generated by constant bit rate (CBR) sessions with sources and destinations uniformly chosen from the population. The aggregate session arrival process is poisson and session durations are exponentially distributed. The total data traffic volume in the network is set to be 60 packets/second and packet size is 512 bytes.

At the outset of the simulation, a fraction of nodes selected uniformly at random from the population are designated as selfish; the remaining nodes behave cooperatively. By varying both the fraction of selfish nodes, we can explore scenarios in which selfish behavior is comparatively rare or widespread. Selfish nodes implement variants of one of two basic selfish strategies—dropping RREQ messages only (**DROP\_REQ**) or dropping RREP messages only (**DROP\_REP**). In both strategies, messages are discarded independently with some dropping probability, which may be either fixed or adaptive, depending on the simulation. For **DROP\_REP**, in addition to dropping RREP messages, the selfish nodes always rebroadcasts RREQs even if it would be possible to generate a reply from cached routing data.

### B. Model of strategic selfishness

In some simulations we model the behavior of a strategic selfish adversary. We assume that the objective of the adversary is to minimize the amount of data forwarded for others while avoiding being classified as selfish by a majority of its neighbors.<sup>8</sup> This model of the adversary raises the question of how the adversary can know how it has been classified by its neighbors. Assuming that cooperative nodes combine some punishment regime with the collection of reputation—as,

<sup>8</sup>One can readily generalize the adversary’s objective as tolerating a selfish classification from at most a fixed fraction of neighbors.

indeed, they must do to present any credible incentive for cooperation—it may be possible for a selfish node to probe its neighbors for evidence of punishment. Alternatively, the system designer may wish to provide explicit notification of selfish classification as a way of allowing falsely accused peers to present some proof of cooperation.

Because we are mainly concerned here with the issue of detection and neither explore nor advocate any specific form of punishment in the present work, we rely on an abstract model of the adversary’s ability to learn about its own classification. Specifically, we assume that the adversary can issue a query to each of its neighbors to determine, without error, that neighbor’s current classification of itself. In our simulation, we implement this query by allowing the simulated adversary to inspect the relevant state of its neighbors directly, i.e. in zero simulated time. Note that this model gives considerably more power to the adversary than exists in practice. For example, an adversary who must probe the system to detect his punishment may not be able to detect his neighbors’ classifications without error.

By querying neighbors, the adversary can implement a straightforward adaptive strategy by periodically querying its neighbors for their classifications and adjusting the probability of dropping messages upwards or downwards accordingly. In our simulation, selfish nodes query its neighbors after every detection round.

We define the adaptive strategy of a selfish node as follows: if more than half of its neighbors classify it as selfish, then it will decrease its dropping probability by a constant  $a$ . Otherwise, it will increase its dropping probability by a constant  $b$ . We consider the case where selfish nodes adapt conservatively—i.e.  $a > b$ . In our simulations, we set  $a = 0.03$  and  $b = 0.01$ .

### C. Detection of fixed selfish strategies

We initially simulate non-adaptive versions of **DROP\_REQ** and **DROP\_REP** strategies, which drop messages with a fixed probability for the duration of the simulation. All selfish nodes behave identically, dropping with the same probability. To evaluate the sensitivity of detection to the aggressiveness of selfish behavior, we varied the dropping probability from 1.0 to 0.1. We conducted simulations with the ANOVA confidence parameter  $\beta$  ranging from 0.1 to 0.9, but present results for  $\beta = 0.4$ , which yields what we felt was the most desirable tradeoff between detection rate and false positive rate. In this experiment, both selfish and cooperative nodes perform detection.

We first consider the **DROP\_REQ** strategy, using a detection window of 400 seconds. Simulation results are

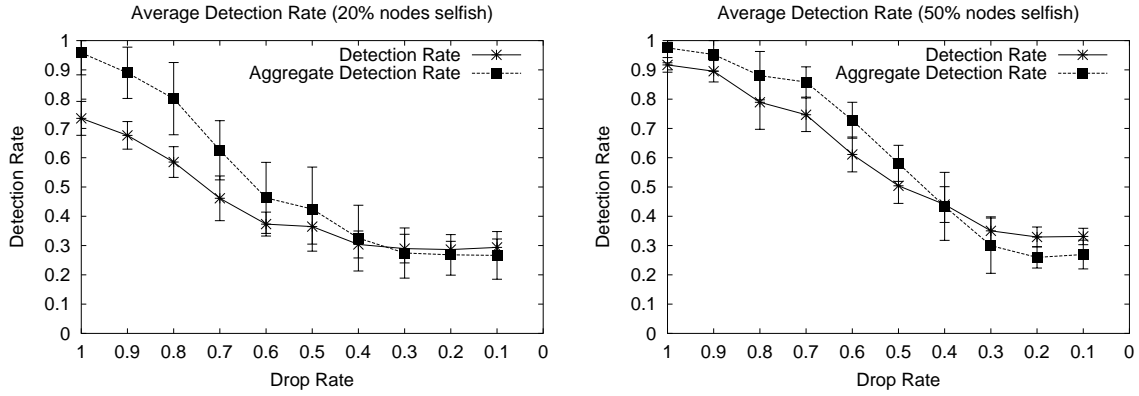


Fig. 3: Detection rate for DROP\_REQ detection.

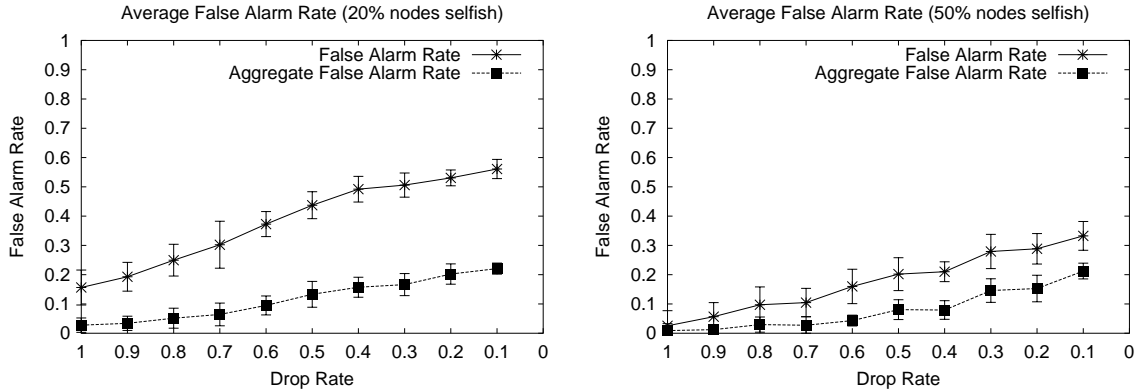


Fig. 4: False alarm rate for DROP\_REQ detection.

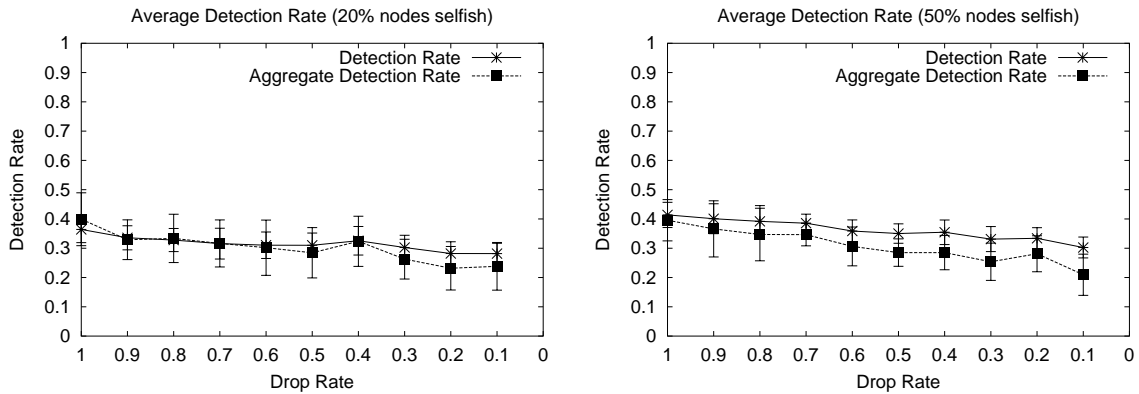


Fig. 5: Detection rate for DROP\_REP detection.

averages of ten executions of the simulation execution, each with a different random seed governing traffic and topology generation. Every simulation run lasts for 1600 seconds.<sup>9</sup> Figures 3 and 4 show the average detection rate and false alarm rate for all cooperative nodes across all detection windows. As expected, the detection rate (false alarm rate) of the test decreases

<sup>9</sup>In every simulation experiment presented here, we measure detection performance only after the after 600 second transient to allow the system to converge to a steady state.

(increases) as selfish nodes become less aggressive.<sup>10</sup>

In anticipation of experiments with strategic adversaries who seek to avoid selfish classification by a majority of neighbors, we find it helpful to define the concept of the *aggregate classification* of a node as the majority opinion of all its neighbors. From this concept follows a natural definition of aggregate

<sup>10</sup>Error bars shown in all plots indicate the sample standard deviation.

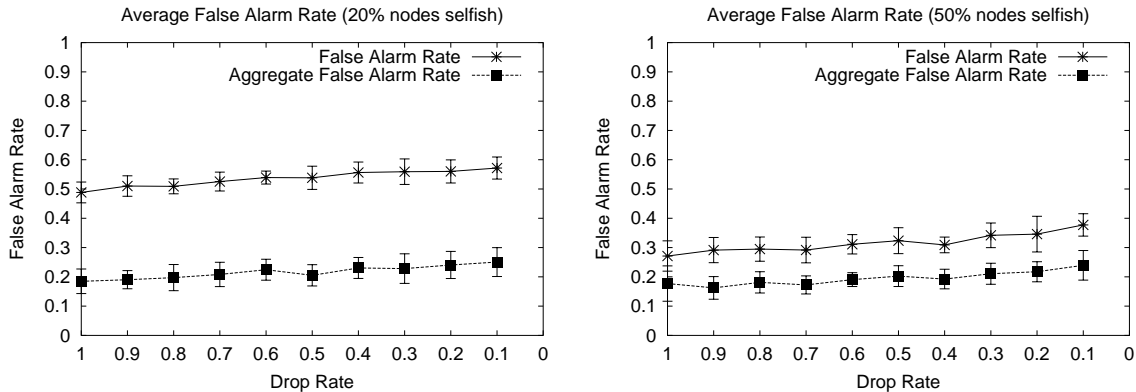


Fig. 6: False alarm rate for DROD\_REP detection.

detection rate—the probability that a selfish node is ‘punished’ by a majority of its neighbors—and an analogous aggregate false alarm rate. Figures 3 and 4 present these aggregate performance measures. In general reliance on collective opinion results in a higher detection rate and lower false alarm rate than that achieved by individual nodes. However, when dropping probability is extremely low, poor detection performance of individual nodes is amplified to produce even worse aggregate performance.

In the case of DROD\_REP, we simulated our detection method using two different detection window sizes: 400 and 9000 seconds. Plots for experiments with window size 9000 seconds have been omitted due to space limitations. Simulation results are averages of ten executions of the simulation, each with a different random seed governing traffic and topology generation. For detection window size 400 (9000), every simulation run lasts for 1600 (10000) seconds. As shown in Figures 5 and 6, with a 400 second detection window, average detection rate and false positive rate were roughly 0.4 and 0.3, respectively, and showed little dependence on dropping probability.

The performance differences in detecting the two types of strategies is a consequence of the fact that AODV—and protocols like it—flood each RREQ throughout the network, creating multiple observable events for each neighbor. RREP messages, in contrast are unicast on select paths, each one involving a small number of nodes. For the majority of nodes not on any identified route, the LRI will be observed to terminate in FSM state 5. Over time, therefore, the rate of RREP-related transitions to states 7 and 8 observed for each neighbor is comparatively low. For example, in a 400 second interval of our simulation, fewer than five such transitions were observed for most nodes. With so little data, the Pearson statistic has very low accuracy. At a window size to 9000 seconds, we observed improved

RREP detection performance, but the resulting protocol was extremely unresponsive.

#### D. Detection of adaptive selfish strategies

To explore the detection of strategic adversaries, we simulate four different scenarios. In addition to simulating our FSM-based detection, we simulate both a totally cooperative network and a network with selfish nodes but no detection. These two scenarios establish upper and lower bounds, respectively, for the amount of data forwarded by selfish nodes. Finally, for comparison, we simulate detection using a Watchdog-like mechanism, which observes implicit acknowledgements of RREQ and RREP messages sent or overhead by the local node, records the frequency of missed acknowledgments, and classifies neighbors as selfish if their rate of dropped messages exceeds some predefined threshold. We vary the fraction of selfish nodes in the network from 0.1 to 0.5. Simulation results are averages of ten executions of the simulation execution, each with a different random seed governing traffic and topology generation.

Figures 7 and 8 show the average dropping probability of selfish nodes and the aggregate false-alarm rate over time.<sup>11</sup> Selfish nodes initially drop requests with probability 0.8. We simulate two versions of Watchdog with threshold values 0.6 and 0.8, with the former being more aggressive at detection than the latter. In the absence of detection, selfish nodes gradually increase their dropping probability until it reaches 1. Under both Watchdog-like and FSM-based detection, selfish nodes are forced to decrease the dropping probability and finally converge to a stable value, which depends on the aggressiveness of the detection algorithm. For FSM-based detection, raising the ANOVA  $\beta$  parameter from 0.3 to 0.4 suppresses selfish behavior below the level achieved by Watchdog with a threshold set at 0.8, but

<sup>11</sup>Time, in these plots, is expressed in units of observation window size, which for these simulations is 100 seconds.

not as low as a very aggressive version of Watchdog (threshold 0.6). As Figure 8 shows, increased aggressiveness results in higher false alarm rates for both algorithms. However, FSM-based method maintains a significantly lower false alarm rate than Watchdog.

Figure 9 shows the the amount of forwarding load by each selfish node, with nodes ranked in decreasing order of amount of data forwarded. When there is no selfish behavior, these nodes do not behave selfishly and thus forward the maximal amount of data. When there is no detection selfish nodes eventually discover that they can drop all RREQs. Although these nodes initially forward some data, over the entire simulation they forward less data than they would in the presence of detection. Under both detection schemes, the number of data packets forwarded by selfish nodes falls between the purely selfish and purely cooperative bounds. Under FSM-based detection, selfish nodes are forced to forward more data packets than under the Watchdog-like method.

Figures 10, 11 and 12 shows the simulation results for detecting RREP dropping with detection window size 400. We observe that the Watchdog-like approach achieves suppresses the dropping probability further than the FSM-based approach but achieves this performance by allowing a higher rate of false alarms. That both methods perform poorly is not surprising: as discussed above, the FSM-based method with small detection window size has poor performance. The Watchdog-like method suffers similarly when data is scarce. These results highlight the intrinsic difficulties of detecting DROP.REP. It is very hard to achieve a high detection rate and maintain a reasonably low aggregate false alarm rate at the same time. When the detection method becomes more aggressive—e.g. by changing the Watchdog threshold from 0.8 to 0.6 or the value of  $\beta$  from 0.3 to 0.4 for FSM-based detection—the selfish nodes are forced to drop the RREP messages with a lower probability. But the aggregate false alarm rate increases in response.

## V. RELATED WORK

Marti, et al. first proposed the Watchdog mechanism to detect faulty and malicious behavior based on dropping data and combined it with a mechanism called Pathrater to avoid routes through such nodes [7]. Buchegger and Le Boudec extended the use of Watchdog-style detection to identify the selfish dropping of data and showed how second-hand reputation information from untrusted nodes could be safely incorporated to improve detection accuracy [14]. Our work differs from these approaches in our focus on detecting selfish routing behavior and our restriction to using only first-hand information.

The use of specification-based detection has become popular in intrusion detection as a way to improve accuracy and reduce false positive rates compared to purely statistical anomaly detection techniques, which use no underlying domain knowledge. Sekar, et al. [15] introduced a general technique using FSM representations of protocols and applied it to detecting misbehavior in wired networks.

Tseng et al. have applied FSM-based techniques to the detection of malicious routing behavior in AODV [16]. Their approach relies on cooperative network monitors to aggregate observations at different locations. Huang and Lee applied anomaly detection [17] and, more recently, specification-based techniques [18] to ad hoc networks for local detection of malicious behavior. Vigna, et al. use a signature-based approach to detect intrusions in AODV [19]. The signature of selfish behavior in this work turns out to be similar to that detected by watchdog.

Our work contrasts with [18] and [16] in that we do not attempt to identify specification violations or transitions to an "alarm state". Similarly to [15], we adopt the FSM description to derive a useful set of features from observed data for statistical analysis. In contrast with [16], we focus on developing a technique that can be implemented on an individual node based only on locally collected data to detect selfish route-avoiding behavior.

SCAN [20] and DICAS [21] have been proposed to mitigate malicious attacks to routing protocols in wireless networks. Our work differs from these protocols in its focus solely on selfish behavior, where the goal is not to prevent misbehavior completely, but to mitigate its performance impact at reasonable cost.

Our work represents a significant improvement over previously published work by the same authors [22], including a completely redesigned detection algorithm and a richer set of simulation results. The most important difference, is a more realistic radio propagation model for simulations. When the realistic radio propagation model is used, the overhearing becomes more unreliable, which results in decreased detection rates and increased false alarm rates of the detection algorithm. The simulation results under realistic model provide a more fundamental understanding of the limitations of detection methods based on overhearing.

## VI. CONCLUSIONS AND FUTURE WORK

In this work, we hypothesized that selfish behavior can be distinguished from cooperative behavior by comparing the statistical behavior of neighbors across multiple local routing instances. We developed a detection technique based on this idea which can readily

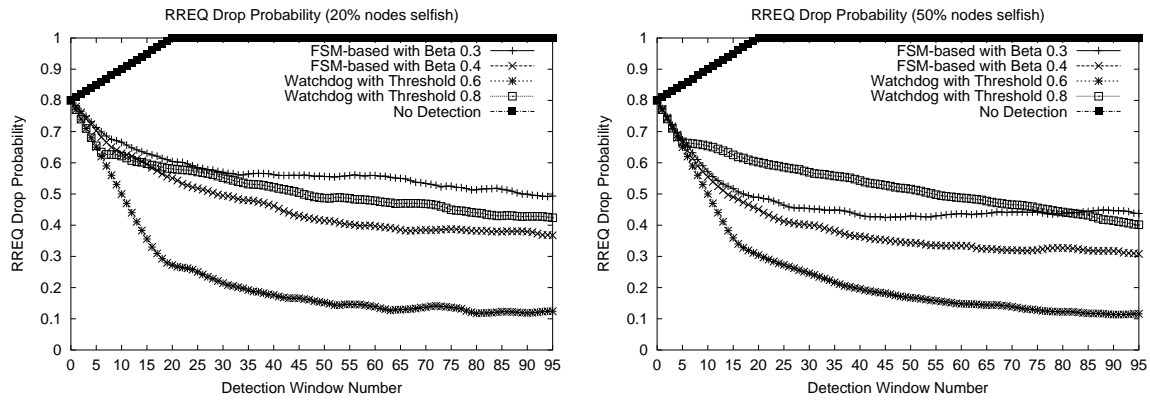


Fig. 7: Selfish dropping probability over time (DROP\_REQ strategy).

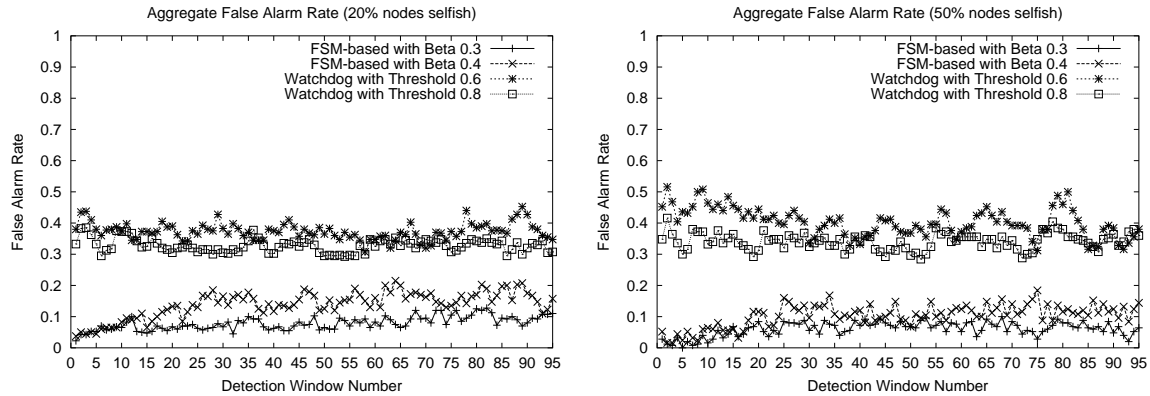


Fig. 8: False alarm rate over time (DROP\_REQ strategy).

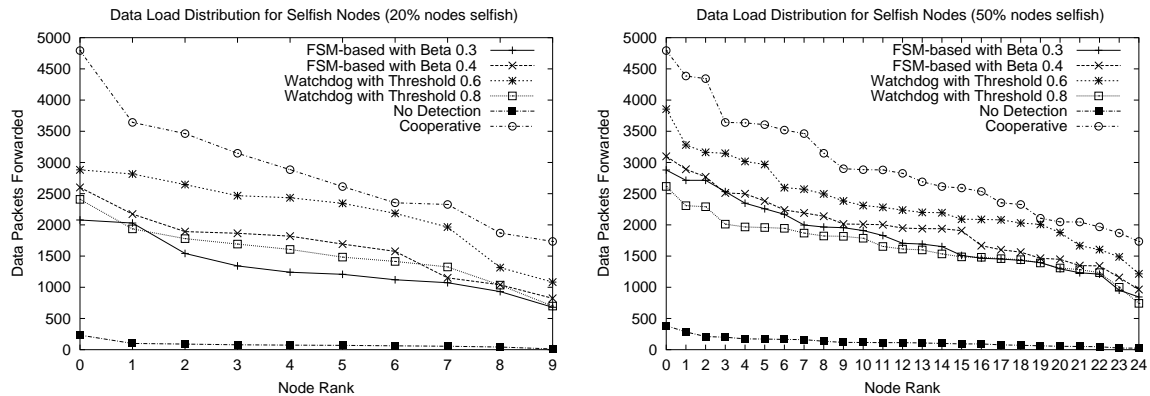


Fig. 9: Data load distribution for selfish nodes (DROP\_REQ strategy).

detect nodes who drop RREQ messages with low false-positive rates even under realistic wireless channel error conditions and against large selfish populations. Against strategic adversaries who seek to avoid detection, our technique forces selfish nodes to accept more routes and thus forward more data than they would in the absence of detection, while maintaining a lower false positive rate than a Watchdog-like approach.

However, in the case of dropped RREP messages, we find that detection is difficult because of the low

rate of route reply events that can be observed by any individual node. Recall that DROP\_REP is a somewhat less effective strategy than DROP\_REQ from a selfish node's point of view due to the risk being selected from cached route information. However, because the strategy is extremely difficult to detect, it becomes an attractive choice for selfish nodes wishing to avoid punishment.

In general, when selfish behavior is difficult to detect, it is appropriate to consider alternative protocol designs

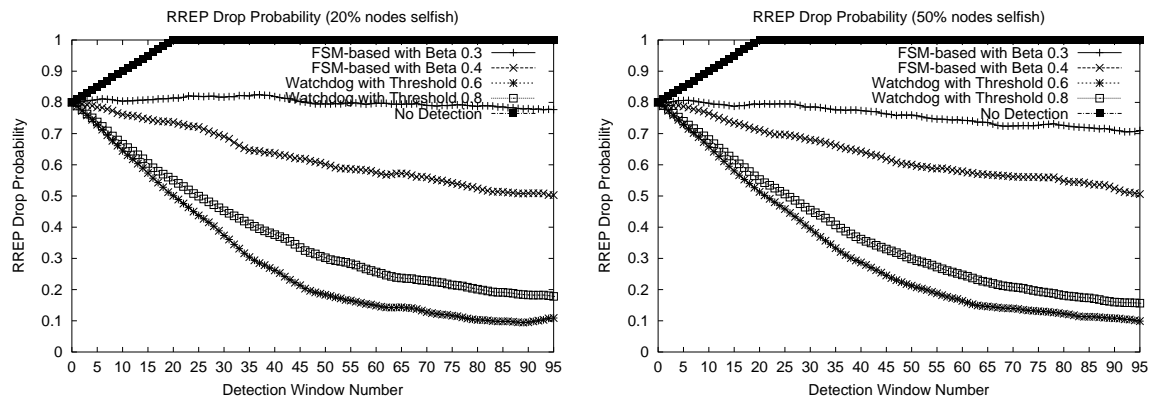


Fig. 10: Selfish dropping probability over time (DROP\_REP strategy).

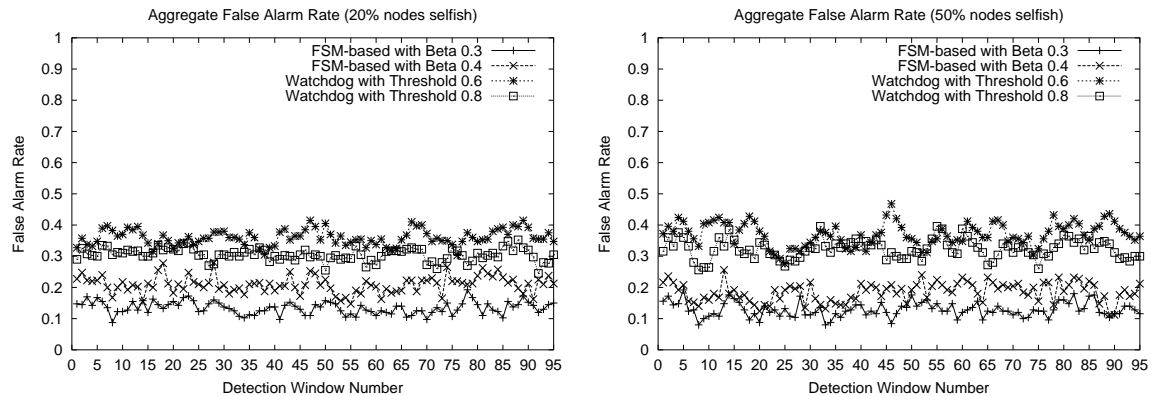


Fig. 11: False alarm rate over time (DROP\_REP strategy).

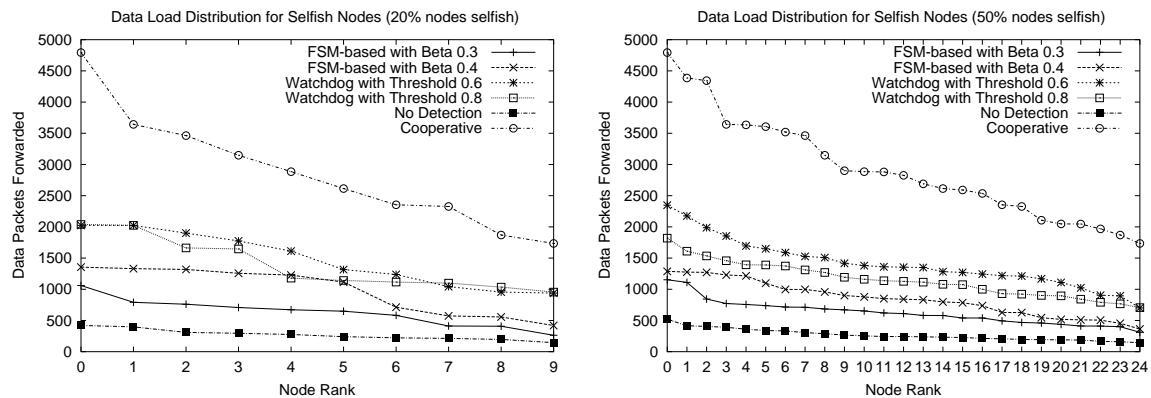


Fig. 12: Data load distribution for selfish nodes (DROP\_REP strategy).

that would render such behavior undesirable. Based on our current work, we believe that modifications to AODV and similar protocols will ultimately be required to remove any incentive to drop route reply messages. We consider such changes an interesting topic for further study. Additionally, open questions remain about whether the punishment phase of reputation mechanisms in ad hoc networks can be similarly decentralized and, in particular, whether punishment might be misinterpreted as evidence of selfishness.

## REFERENCES

- [1] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *MobiCom'05*, Cologne, Germany, 2005.
- [2] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks Journal*, March 2005.
- [3] P. Michiardi and R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Sixth IFIP conference on security communications, and multimedia (CMS 2002)*, Portoroz, Slovenia, 2002.
- [4] B. C. Kim and J. Shaprio, "On the efficacy of detecting and

- punishing selfish peers,” in *Workshop on Internet and Network Economics (WINE)*, Hong Kong, December 2005.
- [5] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, “Sustaining cooperation in multi-hop wireless networks,” in *Networked Systems Design and Implementation (NSDI)*, May 2005.
  - [6] S. Buchegger and J.-Y. L. Boudec, “A robust reputation system for P2P and mobile ad-hoc networks,” in *Second Workshop on the Economics of Peer-to-Peer Systems*, June 2004.
  - [7] S. Marti, T. J. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proceedings MobiCom 2000*, 2000, pp. 255–265.
  - [8] C. E. Perkins and E. M. Royer, “Ad hoc on-demand distance vector routing,” in *Proceedings of the 2nd IEEE workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999, pp. 90–100.
  - [9] K. Pearson, *Philosophical Magazine*, vol. 5, no. 50, pp. 157–176, 1900.
  - [10] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, “Lof: identifying density-based local outliers,” in *ACM SIGMOD*, 2000.
  - [11] W. F. Eddy, A. Mockus, and S. Oue, “Approximate single linkage cluster analysis of large datasets in high dimensional spaces,” *Computational Statistics and Data Analysis*, vol. 23, pp. 29–43, 1996.
  - [12] “<http://www.isi.edu/nsnam/ns/index.html>.”
  - [13] J. Camp, J. Robinson, C. Steger, and E. Knightly, “Measurement driven deployment of a two-tier urban mesh access network,” in *MobiSys*, 2006.
  - [14] S. Buchegger and J.-Y. L. Boudec, “Performance analysis of the CONFIDANT protocol: Cooperation of nodes - fairness in dynamic ad-hoc networks,” in *Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*, Lausanne, CH, June 2002.
  - [15] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou, “Specification-based anomaly detection: A new approach for detecting network intrusions,” in *ACM Computer and Communication Security Conference (CCS’02)*, Washington, DC, USA, November 18–22 2002.
  - [16] C. Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt, “A specification-based intrusion detection system for aodv,” in *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2003.
  - [17] Y. Huang, W. Fan, W. Lee, and P. S. Yu, “Cross-feature analysis for detecting ad-hoc routing anomalies,” in *Proceedings of the 23rd International Conference on Distributed Computing Systems*, May 2003.
  - [18] Y. Huang and W. Lee, “Attack analysis and detection for ad hoc routing protocols,” in *Proceedings of The 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, Sophia Antipolis, France, September 2004.
  - [19] G. Vigna, S. Gwalani, and K. Srinivasan, “An intrusion detection tool for aodv-based ad hoc wireless networks,” in *20th Annual Computer Security Applications Conference*, Tucson, Arizona, December 2004.
  - [20] H. Yang, J. Shu, X. Meng, and S. Lu, “SCAN: self-organized network-layer security in mobile ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 261–273, February 2006.
  - [21] I. Khalil, S. Bagchi, and C. Nina-Rotaru, “DICAS: detection, diagnosis and isolation of control attacks in sensor networks,” in *First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, 2005.
  - [22] B. Wang, S. Soltani, J. K. Shapiro, and P.-N. Tan, “Local detection of selfish routing behavior in ad hoc networks,” in *International Symposium on Parallel Architectures, Algorithms and Networks (I-SPAN)*, Las Vegas, December 2005.