Adaptation of Reputation Management Systems to Dynamic Network Conditions in Ad Hoc Networks

Mohamed Tamer Refaei, Luiz A. DaSilva, *Senior Member*, *IEEE*, Mohamed Eltoweissy, *Senior Member*, *IEEE*, and Tamer Nadeem, *Member*, *IEEE*

Abstract—Reputation management systems have been proposed as a cooperation enforcement solution in ad hoc networks. Typically, the functions of reputation management (evaluation, detection, and reaction) are carried out homogeneously across time and space. However, the dynamic nature of ad hoc networks causes node behavior to vary both spatially and temporally due to changes in local and network-wide conditions. When reputation management functions do not adapt to such changes, their effectiveness, measured in terms of accuracy (correct identification of node behavior) and promptness (timely identification of node misbehavior), may be compromised. We propose an adaptive reputation management system that realizes that changes in node behavior may be driven by changes in network conditions and that accommodates such changes by adapting its operating parameters. We introduce a time-slotted approach to allow the evaluation function to quickly and accurately capture changes in node behavior. We show how the duration of an evaluation slot can adapt according to the network's activity to enhance the system accuracy and promptness. We then show how the detection function can utilize a Sequential Probability Ratio Test (SPRT) to distinguish between cooperative and misbehaving neighbors. The SPRT adapts to changes in neighbors' behavior that are a by-product of changing network conditions, by using the node's own behavior as a benchmark. We compare our proposed solution to a nonadaptive system, showing the ability of our system to achieve high accuracy and promptness in dynamic environments. To the best of our knowledge, this is the first work to explore the adaptation of the reputation management functions to changes in network conditions.

Index Terms—Reputation management, ad hoc networks, cooperation, adaptation, node misbehavior.

1 Introduction

Cooperation among nodes is essential to the survival of an ad hoc network, as it forms the basis for key network functions such as packet forwarding. In particular, nodes are expected to cooperate by forwarding packets on behalf of one another, enabling the delivery of packets over a multihop route. If nodes refuse to cooperate in packet forwarding, end-to-end packet delivery may not be possible. Such uncooperative behavior can greatly degrade network performance and may even result in total communication breakdown. It is important to detect and isolate uncooperative nodes to limit their negative impact on the network performance.

Reputation management systems (RMSs) have been proposed in the literature to promote cooperation and discourage nodes in an ad hoc network from misbehaving with respect to packet forwarding. The goals of a reputation management system are to: evaluate nodes' behavior based on packet forwarding (evaluation), distinguish between cooperative and misbehaving nodes (detection), and appropriately react to nodes according to their behavior (reaction) [1]. It is commonly assumed that reputation management systems at different nodes carry out these functions (evaluation, detection, and reaction) homogeneously across time and space. In other words, the evaluation criteria, detection decision factors, and reactive measures are the same at all nodes at all times. The homogeneous application of reputation decisions does not take into account the dynamics of the environment in which ad hoc networks operate, where local and network-wide conditions change frequently, impacting node behavior. Such behavior may be perceived differently from one node to another. At times, the network conditions are such that a node i that chooses to act cooperatively can successfully do so by forwarding all traffic routed through it (due to moderate traffic activity, favorable channel conditions, etc.). At other times, network conditions are such that node i may not be able to successfully forward traffic routed through it due to adverse conditions (such as high congestion, channel impairments, etc.). Therefore, node behavior may vary from one node to another and from time to time due to changes in local and network-wide conditions, affecting the ability of the reputation management mechanism to distinguish between a node's willful decision not to forward packets (misbehavior) and its inability to do so due to adverse network conditions.

- *M.T. Refaei is with the MITRE Corporation, 7515 Colshire Drive, McLean, VA 22102-7539. E-mail: mrefaei@mitre.org.*
- *L.A. DaSilva is with the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Arlington, VA 22203, and the CTVR, Trinity College Dublin, Dublin 2, Ireland. E-mail: ldasilva@vt.edu.*
- *M. Eltoweissy is with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, 4300 Wilson Blvd., Suite 750, Arlington, VA 22203. E-mail: toweissy@vt.edu.*
- *T. Nadeem is with Siemens Corporate Research, Inc., 755 College Road, Princeton, NJ 08540. E-mail: tamer.nadeem@siemens.com.*

Manuscript received 25 Apr. 2008; revised 13 Feb. 2009; accepted 5 Oct. 2009; published online 29 Jan. 2010.
Recommended for acceptance by Y. Yang.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2008-04-0177.
Digital Object Identifier no. 10.1109/TC.2010.34.

The effectiveness of a reputation management system, which is usually measured in terms of its ability to accurately and promptly distinguish between cooperative and misbehaving nodes, may deteriorate if it does not adapt to network conditions. If the perception of cooperative behavior remains the same under both adequate and adverse network conditions, a node may be incorrectly identified as misbehaving during unfavorable network conditions and inappropriate reaction may be taken against it (e.g., isolation from the network). Hence, in a reputation management system, the evaluation criteria, detection decision factors, and reactive measures should be adaptive across time and space to changes in network conditions. We, herein, refer to such a system as an *adaptive reputation management system.*

In this paper, we focus on enhancing the system's effectiveness through adaptability of the evaluation and detection functions. We rely on two simple premises to guide this adaptation: 1) a sound evaluation of reputation requires the observation of other nodes' behavior for longer periods of time when there is less traffic in the network; and 2) the assessment of other nodes' behavior should be in comparison to one's own behavior. We propose a time-slotted approach for node behavior evaluation. We illustrate how this mechanism can effectively recognize fluctuations in node behavior and how the slot duration impacts the accuracy and promptness of the system in recognizing misbehavior. A slot duration that is too short reduces the system's accuracy, while one that is too long reduces its promptness. We propose a localized approach whereby the slot duration adapts to the traffic activity each node is witnessing.

We, then, introduce an adaptive detection function whereby a cooperative node uses its self-assessed behavior as a benchmark for judging the behavior of other nodes within its local neighborhood. This function takes advantage of the observation that in some environments cooperative nodes within close proximity experience similar network conditions [2], [3]. The detection function devises an SPRT to distinguish between cooperative and misbehaving nodes within each evaluation slot. The SPRT uses packet receiving and packet forwarding events noted for each node within each slot to evaluate node behavior. The SPRT infers network conditions using the node's self-assessed behavior and takes that inference into consideration when evaluating other nodes. A merit of the SPRT is that the number of observations required to accurately evaluate node behavior depends on the local network conditions perceived by the evaluator (hence, localized adaptive detection is possible). Further, the number of observations used in SPRTs is greatly reduced (by 50 percent or more in many cases) when compared to other methods that use a fixed number of observations [4]. This allows for prompt decision making, which limits the scope of damage to the network that can be caused by a misbehaving node. Our goal is to build an effective reputation management system for dynamically changing ad hoc network environments. Such a system must adapt to changes in the network conditions.

Our contribution in this paper is twofold. First, we show that a slotted evaluation function can properly recognize fluctuations in node behavior. We also show that adapting the slot duration is key to maintaining good system accuracy and promptness and illustrate how traffic activity can be used as a point of reference for setting the evaluation slot duration. Second, we show that by using a node's own behavior as a benchmark for judging others' behavior, the detection function can effectively adapt to changes in network conditions. Additionally, we show that an SPRT can accurately and adaptively distinguish between cooperative and misbehaving nodes under different network conditions.

1.1 Scope

In this paper, we are concerned with node behavior with respect to packet forwarding at the network layer. A node can be classified as cooperative or misbehaving. Cooperative nodes do not deliberately drop traffic forwarded through them. We only consider selfish misbehavior in this work, although the concept introduced applies to some types of malicious misbehavior as well, such as black hole and gray hole attacks [5]. Selfishness is intentional misbehavior, where a node chooses not to fully participate in packet forwarding to conserve its own resources. A selfish node may drop some or all packets forwarded through it. A selfish node i is characterized by a selfishness rate, which we denote as s_i, representing the proportion of packets a selfish node will drop among all packets forwarded through it. Selfish nodes typically do not collude with each other or exert additional effort to camouflage their behavior, such as slander attacks [6]. Other classes of outsider attacks or insider node misbehavior at layers other than the network layer are outside the scope of this paper.

We focus on network factors that have a long term effect on a node's ability to successfully forward packets routed through it, which we categorize into two classes: 1) node misbehavior and 2) network environment factors such as congestion at the network layer, contention at the data link layer, and physical communication impairments such as shadowing and path loss. We consider a network environment where detection and isolation of misbehaving nodes is crucial to the survival of the network. Typically, this is the case when the negative impact of node misbehavior on the network performance is dominant compared to deteriorating channel and/or network conditions.

In the next section, we briefly describe the functions of reputation management. In Sections 3 and 4, we discuss adaptability of the evaluation and detection functions. We demonstrate the effectiveness of our system in Section 5. Finally, we discuss related work in Section 6 and conclude in Section 7.

2 Functions of a Reputation Management System

Behavior evaluation, behavior detection, and reaction are the three main functions of reputation management (Fig. 1). The behavior evaluation function adopts an *evaluation metric* and monitors it for nodes whose behavior it is evaluating. The evaluation metric is used to assign each node a *score*, which is a quantification of the node's behavior with respect to packet forwarding. In this paper, we use packet forwarding ratio (the ratio of the number of packets forwarded by a node

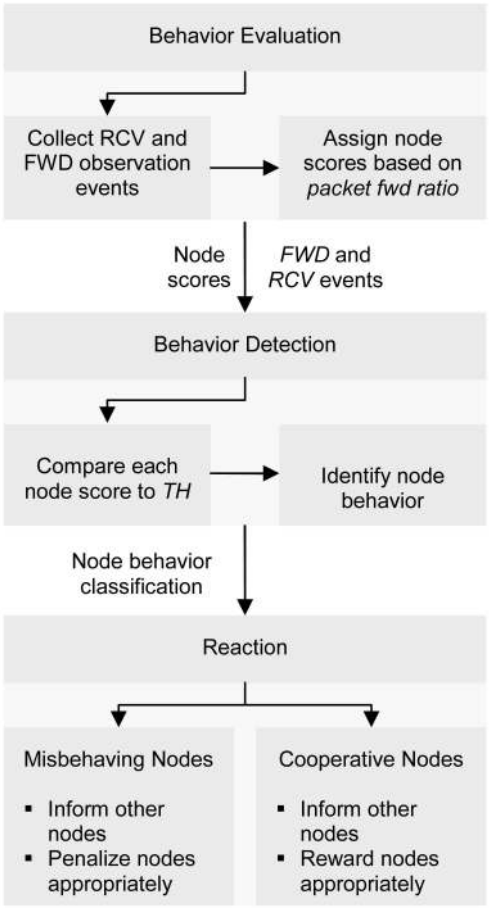

Fig. 1. Functions of reputation management.

to the number of packets routed through it) as the evaluation metric (other metrics have been analyzed in [1]). Hence, each node will monitor packet **forwarding request** events (which we denote as RCV events) and packet **forwarding** events (which we denote as FWD events) for all its neighbors as suggested in [7]. A node's assigned score (i.e., the estimated packet forwarding ratio) ranges from 0 to 1.

The detection function uses the nodes' scores assigned by the evaluation function (and possibly the sequence of events that led to each score) to distinguish between cooperative and misbehaving nodes. A common approach is to compare a node's score to a threshold score TH. For example, in the case of packet forwarding ratio, if the threshold score is 0.8, a node whose score is higher than or equal to 0.8 (i.e., it forwards at least 80 percent of data packets routed through it) is considered cooperative. Nodes with lower scores are considered misbehaving.

Finally, the reaction function takes action against nodes according to their behavior. The reaction taken toward nodes identified by the detection function as misbehaving can be informative, by notifying other nodes about the misbehaving nodes, and/or disciplinary, by penalizing the misbehaving nodes. On the other hand, the reaction taken toward nodes identified as cooperative can reward cooperation by offering these nodes differentiated levels of service according to their level of cooperation.

Table 1 provides a list of notation used throughout the paper.

TABLE 1
Notation Used throughout the Paper

▪ N : The set of nodes in an ad hoc network.
▪ M : The subset of N that contains all misbehaving nodes ($M \subset N$).
▪ G_i : The set of cooperative neighbors of node $i \in N$. $G_i = \{j : j \in N \setminus M \wedge d(i,j) = 1\}$, where $d(i,j)$ is the distance in hops between nodes $i, j \in N$.
▪ $abs(l)$: The absolute value of l.
▪ $a_{j,i,k}$: Score assigned to node $i \in N$ by node $j \in G_i$ during time slot k.
▪ s_i : Selfishness rate of node i.
▪ TH: Threshold value used by non-adaptive reputation management systems to distinguish between the score of a cooperative node from that of misbehaving node.

3 Adaptive Evaluation Function

An adaptive evaluation function must quickly capture changes in node behavior. Node behavior may be impacted by changes in the local and network-wide conditions. An adaptive evaluation function should assign each node a score that reflects its recent behavior.

Our approach (illustrated in Fig. 2) is to perform node behavior evaluation in a time-slotted manner, with slot duration varying according to observed volume of traffic. In

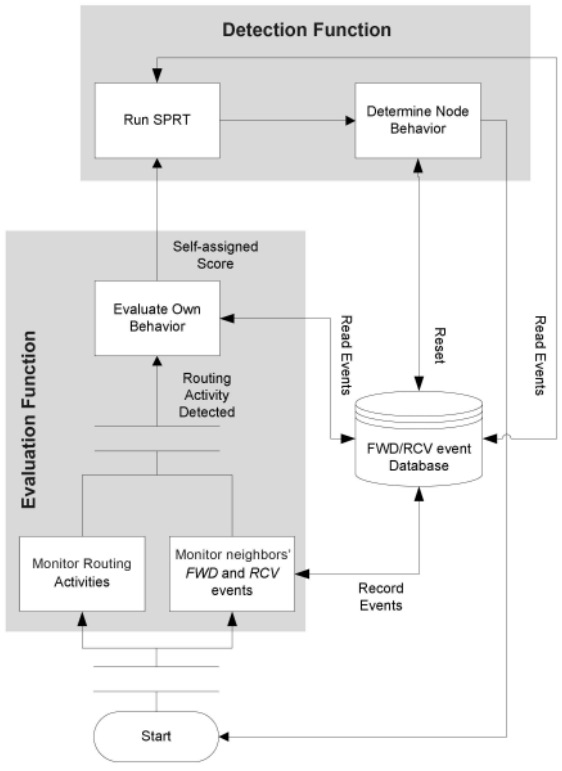

Fig. 2. Flowchart showing the operation of the adaptive evaluation and detection functions.

TABLE 2
Simulation Configuration

Parameter	Value
Simulation Duration	600 seconds
Area	2 km x 2 km
$\|N\|$	64
$\|M\|$	5
Node Range	250 m
Routing Protocol	DSDV
MAC Protocol	CSMA/CA (IEEE 802.11)

this approach, a node's behavior is evaluated and assigned a score (equal to its estimated packet forwarding ratio) by each of its neighbors based on its behavior within each time slot. Given a slot duration τ, a node $i \in \mathrm{N}$ is assigned a score $a_{j,i,k}$ by each of its neighbors $j \in \mathrm{G_i}$ that reflects its behavior within slot $k > 0$ (between times $(k-1)\tau$ and $k\tau$) as perceived by j. The evaluation function also self-assesses node i's own behavior by noting its own RCV and FWD events and assigns itself a score $a_{i,i,k}$ accordingly; this score will be used later by the adaptive detection function.

In a time-slotted behavior evaluation function, the value of the slot duration τ affects the accuracy and the promptness of the reputation management system. This is intuitive since, statistically, τ determines the number of samples that will be considered to estimate a process parameter in a population. In our case, we are trying to estimate the packet forwarding ratio of a node. It is well known that as the sample size increases the accuracy increases. However, collecting too many samples may waste time. We illustrate this point in the example that follows, which is intended as a case study and is followed in the next section by the theoretical foundation of this work (based on an SPRT) and a more comprehensive simulation study.

Consider a network with low traffic activity. Selecting a small τ for such a network reduces the system's accuracy. A short slot may not comprise enough packet forwarding activity to be representative of node behavior, which may result in false positives (cooperative nodes incorrectly identified as misbehaving) or false negatives (misbehaving nodes incorrectly identified as cooperative). On the other hand, consider a network with high traffic activity. Since node behavior evaluation is triggered at the end of each slot, selecting a value of τ that is too high may impact the system's promptness (i.e., timely identification of node behavior). Accordingly, the slot duration must adapt according to the traffic activity in the network. As the traffic activity increases, the slot duration should decrease. There is also a trade-off between slot duration and overhead. Decreasing the slot duration in a network with high traffic achieves better promptness but may also result in increased computational overhead.

We illustrate the relationship between traffic activity and the value of τ through a sensitivity analysis conducted using ns-2 simulations (the simulation parameters are shown in Table 2). We consider a network of $|\mathrm{N}| = 64$ nodes and $|\mathrm{M}| = 5$ misbehaving nodes. We evaluated the accuracy and promptness of node behavior evaluation using different values of τ

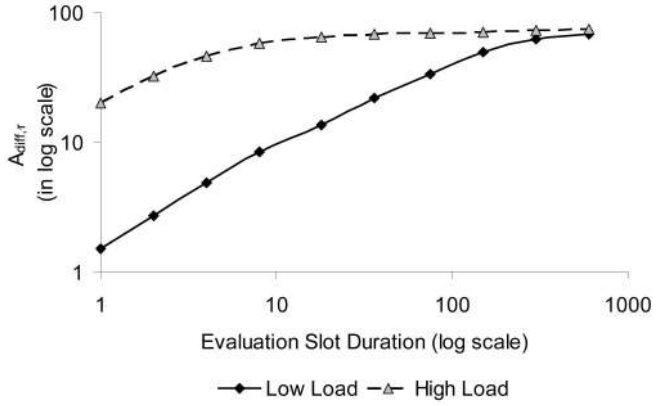

Fig. 3. System accuracy: comparing $A_{\mathrm{diff},\tau}$ for high and low traffic activities as a function of evaluation slot duration. The values in the y-axis are expressed as percentages (i.e., the ratio is multiplied by 100).

in $D = \{1, 2, 4, 8, 18, 36, 75, 150, 300, 600\}$ seconds. We measure accuracy based on the difference between the average score assigned to misbehaving nodes by their cooperative neighbors and that assigned to cooperative nodes by their cooperative neighbors. We denote the difference as A_{diff}, which we calculate as $A_{\mathrm{diff}} = abs(A_{coop} - A_{misb})$, where

$$A_{coop} = \begin{cases} \dfrac{1}{|\mathrm{N}| - |\mathrm{M}|} \displaystyle\sum_{i \in N \setminus M} A_i, & \text{if } |M| < |N|, \\ 0, & \text{otherwise,} \end{cases}$$

$$A_{misb} = \begin{cases} \dfrac{1}{|\mathrm{M}|} \displaystyle\sum_{i \in M} A_i, & \text{if } |M| > 0, \\ 0, & \text{otherwise,} \end{cases}$$

$A_i = \frac{1}{|\mathrm{G_i}|} \sum_{j \in \mathrm{G_i}} A_{j,i}$, $A_{j,i} = \frac{1}{s} \sum_{k=1}^{s} a_{j,i,k}$, and s is the total number of evaluation slots ($s = \text{Simulation Duration}/\tau$). For each $\tau \in D$, we calculate A_{diff}, which we denote as $A_{\mathrm{diff},\tau}$. Note that $0 \leq A_{\mathrm{diff},\tau} \leq 1$ and that as the value of $A_{\mathrm{diff},\tau}$ decreases it becomes more difficult for the detection function to distinguish between cooperative and misbehaving nodes, which increases false positives as well as false negatives (i.e., decreases accuracy). We use two different levels of traffic activity: high (200 short-lived flows with randomly selected sources and destinations) and low (20 short-lived flows with randomly selected sources and destinations). Fig. 3 indicates that as the value of τ increases, the value of $A_{\mathrm{diff},\tau}$ also increases. Increasing the value of τ allows for more packet forwarding activity to take place within a slot so as to be representative of node behavior. Note the clear difference between low and high traffic activities. For the high traffic activity, high values of $A_{\mathrm{diff},\tau}$ occur even with small slot durations, as the number of interactions within a slot is higher. If the system, for instance, requires $A_{\mathrm{diff},\tau} \geq 0.25$ to achieve a target accuracy level, this corresponds to slot duration $\tau \geq 2$ seconds for the high traffic activity case and $\tau \geq 75$ seconds for the low traffic activity case.

The trade-off to the increase in accuracy as a result of increasing the value of τ is the decrease in promptness. Here, we define promptness as the earliest time A_{diff} can achieve a target accuracy level. Fig. 4 plots the promptness of the system with the accuracy level used earlier ($A_{\mathrm{diff},\tau} \geq 0.25$)

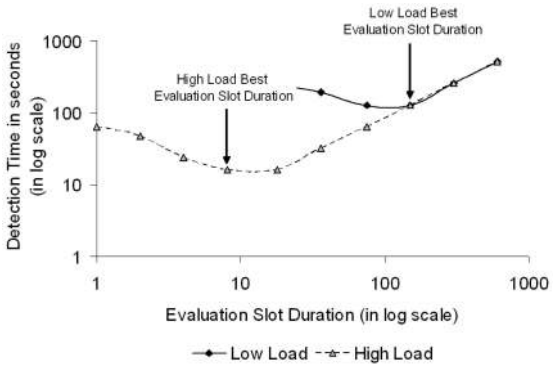

Fig. 4. System promptness: misbehavior detection time for high and low traffic activities as a function of evaluation slot duration.

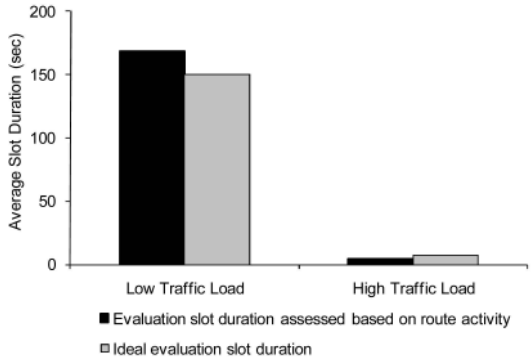

Fig. 5. Comparing the ideal value of τ to the average value of τ assessed based on route activity.

denoting promptness of a particular value of $\tau \in D$ as the earliest time when $A_{\text{diff},\tau} \geq 0.25$. The figure shows that unnecessarily increasing the value of τ may decrease promptness (noting that behavior evaluation is triggered at the end of each evaluation slot, followed by triggering of the detection function as shown in Fig. 1). It also shows that there is a particular value of τ where the required accuracy level is reached fastest. This is the ideal slot duration where enough packet forwarding activity is present at the earliest time to distinguish between the behavior of cooperative nodes and that of misbehaving nodes at the required accuracy level. Shorter slot durations do not take into consideration enough packet forwarding activity, while longer slot durations unnecessarily delay node evaluation, both resulting is lower promptness. From the figure, the slot duration in which misbehaving nodes are detected fastest at the required accuracy level is $\tau = 8$ seconds for the high traffic activity case and $\tau = 150$ seconds for the low traffic activity case.

We propose an adaptive approach whereby the value of τ adapts automatically to the traffic activity. We consider route activity as a measure of traffic activity. The frequency of route establishment, route update, and route expiration events (what we mean by route activity) increases as traffic activity increases (otherwise, routes cannot be established, end-to-end packet delivery is hindered, and the network becomes unstable). Whenever a route establishment, route update, or a route expiration event occurs, each node that becomes aware of the update (the route event affects one or more fields in the node's routing table) triggers the behavior evaluation function to evaluate its neighbors by assigning each a score based on the FWD and RCV events noted for each since the last evaluation. For a given node i, if we consider routing activities to be indexed by r, where $r = 0$ corresponds to the first routing activity noted by the node. If we consider $t_{i,r}$ to be the time of arrival of the route request, update, or expiration message, then the value of τ noted by node i at time t (which we denote as $\tau_{i,t}$) can be represented as

$$\tau_{i,t} = \begin{cases} t_{i,r} - t_{i,r-1}, & \text{if } r > 0, \\ t_{i,r}, & \text{otherwise.} \end{cases}$$

The distribution of the interarrival time of routing messages at a given node will induce the distribution for τ at that node. Hence, the value of τ differs from one node to another and from time to time according to the traffic activity each node is witnessing (e.g., a node located in the center of the network will typically experience higher traffic activity compared to nodes located close to the network boundaries).

We demonstrate our approach for the adaptation of τ using ns-2 simulations. In Fig. 5, we show the average slot duration for periods of high and low traffic compared to the ideal values obtained from Fig. 4. Fig. 6 shows, for 10 different traffic activity levels, how the increase in traffic activity decreases the value of τ.

4 Adaptive Detection Function

An adaptive detection function is able to distinguish between cooperative and misbehaving nodes under different network conditions. Consider first a nonadaptive detection function. Such a detection function uses a fixed threshold TH whose value is identical for all nodes in the network (as described in Section 2). TH is usually assigned to reflect acceptable node behavior under normal network conditions. However, if network conditions change (e.g., due to a sudden deterioration of the channel), the selected value of TH may no longer be appropriate, resulting in inaccuracies in the detection function. An adaptive detection function, on the other hand, adapts its criteria for cooperative behavior according to the state of the network.

Our adaptive detection function (described in Fig. 2) makes a decision about node behavior within each behavior evaluation slot based on an SPRT. We consider each node's self-assigned score as well as the FWD and RCV observation events it noted for each of its neighbors in the SPRT. In the next sections, we provide an overview

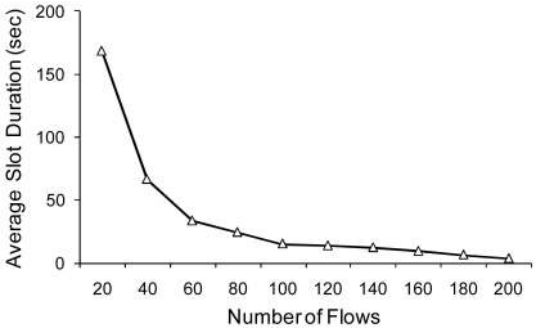

Fig. 6. Average value of τ assessed based on route activity for 10 different traffic activity levels.

of SPRT, introduce the SPRT for judging node behavior, and then illustrate how the test adapts to changes in network conditions.

4.1 Overview of SPRT

Consider two hypotheses H_1 and H_0, where either H_1 or H_0 is true but not both. In traditional hypothesis testing, one of the decisions (H_1 or H_0) is made after a random sample is observed. In some cases, the evidence in the random sample observed may be enough to make the decision while in other cases it may not. Traditional hypothesis testing mechanisms make a decision either way. Sequential probability ratio testing defers the decision if the evidence in the random sample observed is not sufficient to make an accurate decision and requests more observations. The test continues until the evidence is strong enough to make a decision [4].

SPRTs consider two conditional probabilities, $P[x|H_1]$ and $P[x|H_0]$. To decide whether H_1 or H_0 is true we make a sequence of observations $x_1, x_2, \ldots$. For each observation $x_n, n \geq 1$, if $P[x_n|H_0] \neq 0$ we calculate the ratio $\frac{P[x_n|H_1]}{P[x_n|H_0]}$ and accumulate the value $T_n = T_{n-1} * \frac{P[x_n|H_1]}{P[x_n|H_0]}$, where $T_0 = 1$. We then examine the value of T_n. If T_n is large, it implies that the sequence of observations $x_1, \ldots, x_n$ made so far are more likely to have been generated under H_1 than under H_0. If T_n is small, the converse is true. If T_n is not sufficiently small or large to choose between H_1 and H_0, we make another observation x_{n+1}.

In selecting between H_1 and H_0, it is possible to erroneously decide that H_1 is true while in reality H_0 is true (a false positive), or that H_0 is true while in reality H_1 is true (a false negative). To limit the probability of false positives to α and that of false negatives to β, we select two threshold values A and B, with $B < A$. After making a sequence of observations $x_1, \ldots, x_n$, a decision is made that H_1 is true if $T_n \geq A$, or that H_0 is true if $T_n \leq B$ and the test terminates. We make an additional observation x_{n+1} if $B < T_n < A$. It was shown in [4] that the values of A and B are bounded by $A \leq \frac{1-\beta}{\alpha}$ and $B \geq \frac{\beta}{1-\alpha}$ and that by using values of $A = \frac{1-\beta}{\alpha}$ and $B = \frac{\beta}{1-\alpha}$ the test provides adequate level of precision.

4.2 SPRT for Judging Node Behavior

At the end of each evaluation slot k, the behavior evaluation function of node $j \in \mathrm{N} \backslash \mathrm{M}$ passes to the detection function the list of scores assigned to each neighbor and the sequence of events that led to each score (RCV and FWD observation events noted for each neighbor $i \in \mathrm{G_j}$ during time slot k). The behavior evaluation function also passes the node's self-assigned score $a_{j,j,k}$ to the detection function. The detection function analyzes the list of RCV and FWD events and infers observations DRP for each packet routed through the node but not forwarded, denoting that the packet was most likely dropped.[1]

The SPRT considers two hypotheses H_1 and H_0. H_1 is the hypothesis that a given node is misbehaving while H_0 is the hypothesis that it is cooperative. Let $T(j,i)_n$ be an evaluation of node i's behavior by node $j \in \mathrm{G}_i$ within the current time slot. $T(j,i)_n$ considers all FWD and DRP observation events noted for node i within the current evaluation slot. Upon encountering an event x_n that is noted for node i by node j, node j then evaluates

$$T(j,i)_n = \begin{cases} T(j,i)_{n-1} * \dfrac{P[FWD|H_1]}{P[FWD|H_0]}, & \text{if } x_n = FWD, \\ T(j,i)_{n-1} * \dfrac{P[DRP|H_1]}{P[DRP|H_0]}, & \text{if } x_n = DRP, \end{cases}$$

where $T(j,i)_0 = 1$. If $T(j,i)_n \geq A$, the test concludes that H_1 is true and that node i has been misbehaving during the last n events. If $T(j,i)_n \leq B$, the test concludes that H_0 is true and that node i has been acting cooperatively during the last n events. The outcome of the test is recorded and the test is restarted for another evaluation round starting at event x_{n+1}. At the end of the evaluation slot, all events have been considered and a number of outcomes have been recorded. All recorded test outcomes are considered to decide upon node i's behavior. We consider node i misbehaving if most test outcomes concluded H_1. We consider node i cooperative otherwise.

The SPRT considers α and β to be configurable system parameters that represent the level of false positives and false negatives that are tolerable by the system. From the configured values of α and β, the threshold values of A and B are set (as suggested [4]) to $A = \frac{1-\beta}{\alpha}$ and $B = \frac{\beta}{1-\alpha}$. Note that the use of TH in nonadaptive reputation management systems can be related to the use of A in our adaptive system since both are considered the boundary toward misbehavior. Their values, however, cannot be equated since A is a probability ratio (calculated based on the false positives and false negatives the system can tolerate) and TH is a score.

The values of $P[FWD \mid H_1]$, $P[FWD \mid H_0]$, $P[DRP \mid H_1]$, and $P[DRP \mid H_0]$ depend on the probability of packet drops in the network due to misbehavior and due to network environment factors (such as congestion at the network layer, contention at the data link layer, physical communication impairments such as fading, etc.), which we denote as $P_{misbehavior}$ and P_{drop}, respectively. The conditional probabilities can then be obtained as shown in [8]:

- $P[FWD \mid H_1] = (1 - P_{drop}) * (1 - P_{misbehavior})$.
- $P[FWD \mid H_0] = (1 - P_{drop})$.
- $P[DRP \mid H_1] = 1 - (1 - P_{drop}) * (1 - P_{misbehavior})$.
- $P[DRP \mid H_0] = P_{drop}$.

Note that according to the definition of $P[FWD \mid H_1]$ and $P[FWD \mid H_0]$, the probability ratio will only depend on $P_{misbehavior}$ when a FWD event is witnessed (since $\frac{P[FWD|H_1]}{P[FWD|H_0]} = 1 - P_{misbehavior}$).

The value of $P_{misbehavior}$ can be viewed as a network parameter that is set according to the network objective. For example, setting the value of $P_{misbehavior}$ to 50 percent indicates that the network aims to detect all misbehaving nodes $i \in \mathrm{M}$ where $s_i \geq 50\ percent$. This also implies that the network can tolerate some false negatives when it comes to detecting misbehaving nodes whose $s_i < 50\ percent$. Setting the value of $P_{misbehavior}$ is analogous to setting the security mode of an Intrusion Detection System (IDS). Setting the security mode of an IDS too high is likely to expose the network to fewer threats (i.e., low false

1. Node j assigns a time-out for each RCV event of a packet routed through node during which an FWD event of the same packet must be noted. Otherwise, the packet is judged to have been dropped by node i.

negatives) but may also trigger a large number of false positives. Setting the security mode of an IDS too low may expose the network to a lot of threats (i.e., high false negatives) but reduces the number of false positives. The ideal setting may vary depending on the level of the threat that the network is likely to be exposed to and the likely outcome if any such threats prevail. Much like the security mode of an intrusion detection system, the value of $P_{misbehavior}$ should be set according to the perception of the level of misbehavior that is likely to exist in the network and its impact on each node. We will discuss setting the value of P_{drop} in the next section.

4.3 Adaptation of the SPRT

The goal of our adaptive reputation management system is to adapt node behavior evaluation and detection to dynamic changes in network conditions. Our goal is not to adapt the system to changes in the threat level that the network is exposed to. Hence, the values of α, β, and $P_{misbehavior}$ need not be adaptive but the value of P_{drop} needs to adapt to reflect current network conditions.

Consider a cooperative node $j \in \mathrm{N}\backslash\mathrm{M}$ that is evaluating the behavior of its neighbor $i \in \mathrm{G_j}$. The time-slotted evaluation function of node j has evaluated node j's own behavior at the current evaluation slot k based on the ratio between the number of its own RCV and FWD events (i.e., the ratio of the number of packets node j forwarded to the number of packets routed through it) and assigned itself a score $a_{j,j,k}$. The evaluation function then passes the score to the detection function.

Our detection function makes use of the observation that cooperative nodes within close proximity may experience similar network conditions [2], [3]. If node j's cooperative behavior under the current network conditions granted it a score of $a_{j,j,k}$ then the value of $a_{j,j,k}$ can be used to estimate the expected behavior of other cooperative nodes within node j's local neighborhood, which operate under similar local network conditions. Additionally, $1 - a_{j,j,k}$ can be used to estimate the probability of packet drop due to network environment factors in the node's local neighborhood (i.e., packets dropped despite the node's intention to act cooperatively). Note that both estimates continuously adapt to changes in the node's local network conditions. In our detection function, we use $P_{drop} = 1 - a_{j,j,k}$. The adaptation of the detection function is realized as follows:

- The SPRT at node j will move faster toward threshold A for any DRP event observed for node i if node j experienced a low value of P_{drop} during slot k. Hence, the number of observations needed to make an H_1 decision will be small in case of a low value of P_{drop}.
- The SPRT at node j will move more slowly toward threshold B for any FWD event observed for node i if node j experienced a low value of P_{drop} during slot k. Hence, the number of observations needed to make an H_0 decision will be large for a low value of P_{drop}.
- The SPRT at node j will move more slowly toward threshold A for any packet drop event observed at node i if node j experienced a high value of P_{drop} during slot k. Hence, the number of observations needed to make an H_1 decision will be large in case of a high value of P_{drop}.
- The SPRT at node j will move faster toward threshold B for any FWD event observed for node i if node j experienced a high value of P_{drop} during slot k. Hence, the number of observations needed to make an H_0 decision will be small in case of a low value of P_{drop}.

TABLE 3
Summarizing the Parameters of the Adaptive Reputation Management System

Parameter	Value
α	A system parameter set according to the tolerable level of false positives.
β	A system parameter set according to the tolerable level of false negatives.
A	$A = \frac{1-\beta}{\alpha}$ as suggested in [4].
B	$B = \frac{\beta}{1-\alpha}$ as suggested in [4].
$P_{misbehavior}$	A system parameter set according to the tolerable level of misbehavior.
P_{drop}	Calculated locally by each node by observing its own behavior under the current local network conditions.

The SPRT offers a systematic approach to select the thresholds and adapt node evaluation to network conditions. As indicated above, adaptation is accomplished by weighing events (i.e., DRP or FWD) differently depending on the network conditions. We envision that it is also possible to devise a heuristic approach (as opposed to the systematic approach offered by SPRTs) that varies TH according to the network conditions.

Let us briefly consider what happens if a selfish node adopts the same reputation management mechanisms proposed here. If a selfish node considers packets dropped due to its selfish nature as DRP events during self-assessment, its self-assigned score will most likely be lower than that of its cooperative neighbors. Hence, the self-assigned score will not accurately reflect the behavior of a cooperative node within the selfish node's locality, which may increase false negatives. On the other hand, a selfish node may chose to ignore the effect of its selfish behavior during self-assessment by assuming DRP events due to the node's selfishness as FWD events. This results in a self-assigned score that reflects more closely the behavior of a cooperative node within the selfish node's locality.

Table 3 summarizes the parameters used in our adaptive reputation management system and how the value of each is set. We also summarize the advantages of using SPRTs in node behavior evaluation in the following points:

1. *Ease of implementation.*
2. *Sequential evaluation*: The number of observations (i.e., FWD and DRP events) required to evaluate node behavior need not be determined in advance.

TABLE 4
Simulation Configuration

Parameter	Value
Simulation duration	1000 seconds
Area	500x500 meters
$\|N\|$	25
Node Range	250 meters
Number of flows	1000 UDP flows
Flow Data Rate	4 Kbits/sec
Max packets per flow	10 packets
Routing Protocol	DSDV
MAC Protocol	CSMA/CA (IEEE 802.11)

This property of SPRTs suits well the dynamic nature of ad hoc networks where the rate and the *significance* of such observations may change spatially and temporally.

3. *Configurable accuracy*: The specified degree of accuracy can be set by configuring α and β.
4. *Prompt decision making*: The number of observations used to evaluation node behavior (at the configured accuracy level) using an SPRT is greatly reduced when compared to other methods that use a fixed number of observations [4]. This allows for prompt decision making, which limits the scope of damage to the network that can be caused by misbehaving nodes.

5 Demonstration of Effectiveness

In this section, we demonstrate the effectiveness of our adaptive system under different network conditions. We also aim to show the shortcomings of nonadaptive approaches. We consider network environments with varying rates of change in network conditions as well as varying levels of misbehavior in the network.

We consider two types of network environments in our evaluation: static and dynamic. An environment is static if the network conditions in that environment do not change throughout one run of the simulation. A dynamic environment is one where the network conditions change frequently. Further, our simulations account for the impact of different levels of misbehavior in the network. Two parameters control the level of misbehavior in the network: 1) the number of misbehaving nodes, and 2) the selfishness rate of each misbehaving node. By devising network scenarios that use different ratios of misbehaving to well-behaved nodes and different selfishness rates, we can model varying levels of misbehavior in the network.

We consider three sets of network scenarios denoted as SC-1, SC-2, and SC-3. All three scenarios share the simulation parameters shown in Table 4. In SC-1, we consider all nodes to be cooperative (i.e., $|M| = 0$). In SC-2, we consider 20 percent of the nodes in the network to be selfish (i.e., $|M| = 5$). In SC-3, we consider 40 percent of all nodes in the network to be selfish (i.e., $|M| = 10$). The value of s_i for a misbehaving node in SC-2 and SC-3 is randomly selected from the set $\{50\ percent, 75\ percent, 100\ percent\}$.

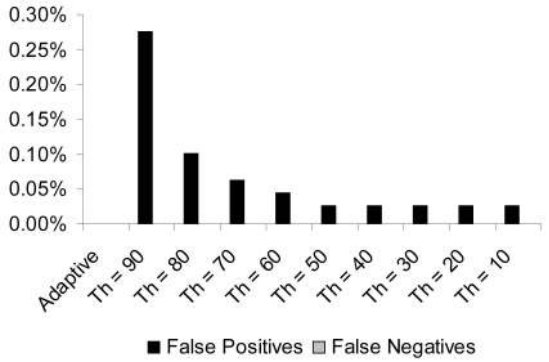

Fig. 7. SC-1—SE-1: All nodes are cooperative. Cooperative nodes can forward almost all data packets routed through them.

We consider a configuration of our adaptive reputation management system that uses $P_{misbehavior} = 50\ percent$. As mentioned before, this indicates that the network is keen to detect all misbehaving nodes whose selfishness rate is greater than or equal to 50 percent (which also implies that the network can tolerate some false negatives when it comes to detecting misbehaving nodes whose selfishness rate is less than 50 percent). We consider $\alpha = 10\ percent$ and $\beta = 10\ percent$ to be the target accuracy level of the system. For the nonadaptive system, we consider a range of values for TH. All our results are averaged over 20 runs. Each run considers a different randomly generated topology.

5.1 Static Environment

We show in this section that the adaptive reputation management system can operate effectively under different network conditions, while the settings of the nonadaptive reputation management system must be customized for a given network condition for the system to be effective. We consider three scenarios with respect to the network conditions:

- SE-1: The network conditions are such that a cooperative node can forward almost all packets forwarded through it.
- SE-2: The network conditions are such that a cooperative node can forward at most 80 percent of all packets forwarded through it.
- SE-3: The network conditions are such that a cooperative node can forward at most 60 percent of all packets forwarded through it.

Since the environment is static, we let the adaptive approach evaluate node behavior using a fixed value of τ. We compare against a nonadaptive approach by assessing the false positives and false negatives. Our comparison considers different values of TH for the nonadaptive approach.

The results in Figs. 7, 8, 9, 10, 11, 12, 13, 14, and 15 show that the adaptive system can operate under all conditions considered with low false positives and false negatives. The maximum false positives over all scenarios was 11.60 percent (within reasonably range of α), and the maximum false positives was at 3.12 percent (bounded by β). This indicates that the adaptive approach self-adapts its criteria for identifying cooperative and misbehaving behaviors according to the network condition it is operating in. On the other

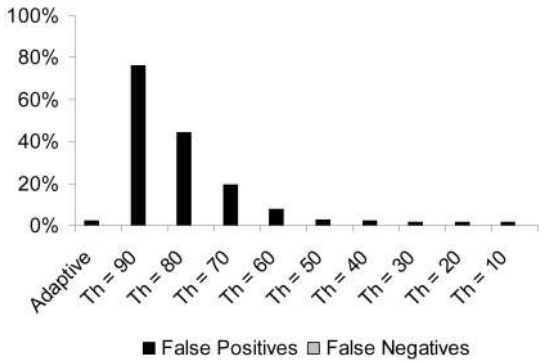

Fig. 8. SC-1—SE-2: All nodes are cooperative. Cooperative nodes can forward at most 80 percent of data packets routed through them.

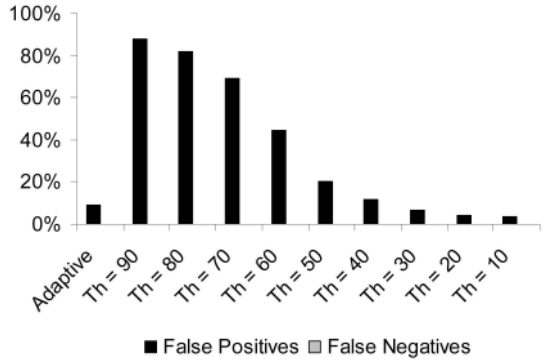

Fig. 9. SC-1—SE-3: All nodes are cooperative. Cooperative nodes can forward at most 60 percent of data packets routed through them.

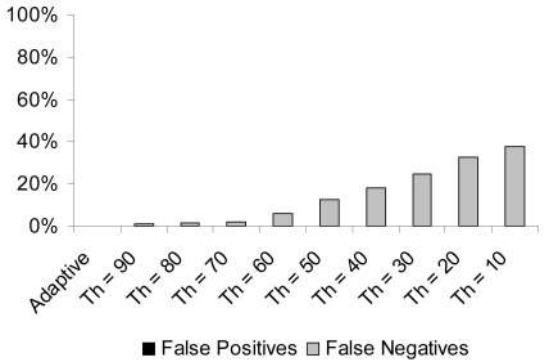

Fig. 10. SC-2—SE-1: 20 percent of all nodes are selfish. Cooperative nodes can forward almost all data packets routed through them.

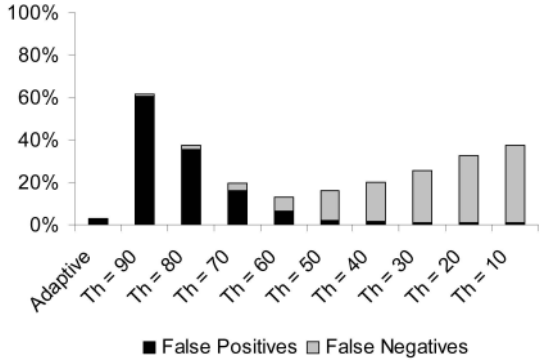

Fig. 11. SC-2—SE-2: 20 percent of all nodes are selfish. Cooperative nodes can forward at most 80 percent of data packets routed through them.

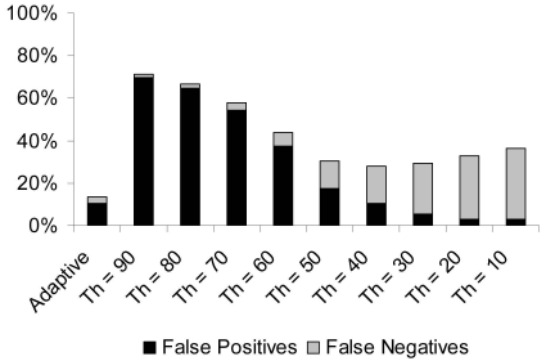

Fig. 12. SC-2—SE-3: 20 percent of all nodes are selfish. Cooperative nodes can forward at most 60 percent of data packets routed through them.

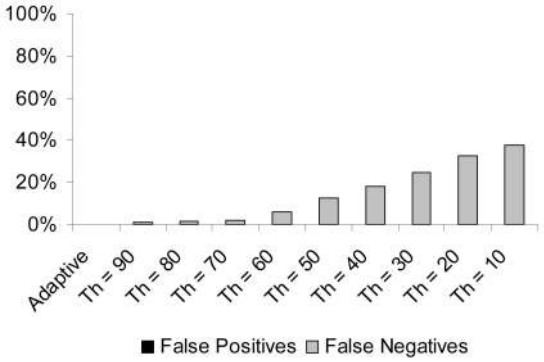

Fig. 13. SC-3—SE-1: 40 percent of all nodes are selfish. Cooperative nodes can forward almost all data packets routed through them.

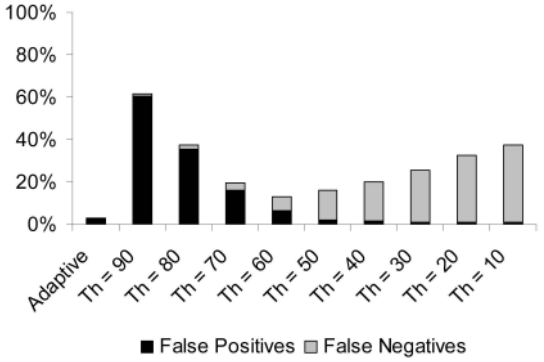

Fig. 14. SC-3—SE-2: 40 percent of all nodes are selfish. Cooperative nodes can forward at most 80 percent of data packets routed through them.

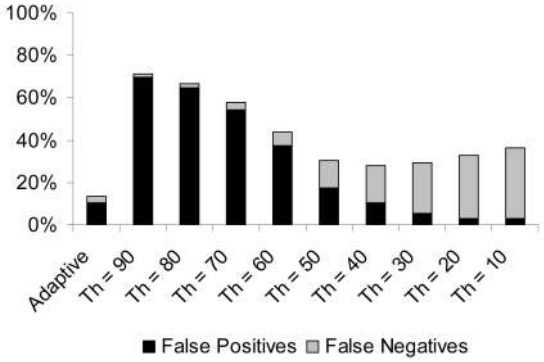

Fig. 15. SC-3—SE-3: 40 percent of all nodes are selfish. Cooperative nodes can forward at most 60 percent of data packets routed through them.

TABLE 5
Ideal TH for the Nonadaptive Approach

Ideal TH	SC-1	SC-2	SC-3
SE-1	$TH \leq 50$	$TH = 90$	$TH = 90$
SE-2	$TH = 10$	$TH = 60$	$TH = 60$
SE-3	$TH = 10$	$TH = 40$	$TH = 40$

hand, the nonadaptive approach requires the value of TH to be customized to reflect the network conditions where it operates. If the value of TH is not set correctly the system will yield high false positives and/or false negatives. For example, if TH is set to 0.90 in SC-1—SE-3, the system will yield close to 90 percent false positives. Table 5 lists the ideal values of TH for the nonadaptive approach in all considered scenarios. We consider the ideal value as the one that yields the lowest average false positives and false negatives. Note how the ideal value of TH changes as the network conditions, as well as the level of misbehavior in the network, vary.

5.2 Dynamic Environment

This section presents results regarding the resilience of the adaptive reputation management system to variable network conditions. We show that the adaptive reputation management system can operate effectively under changing network conditions, achieving false positives and false negatives that are close to the values of α and β, respectively.

We let τ adapt according to the traffic activity as explained in Section 3. We modified the ns-2 simulator to create simulation scenarios where network conditions fluctuate consistently. We consider the shadowing propagation model, which consists of two parts: the path loss model and the shadowing deviation model [9]. The path loss model predicts the received power of a signal as it propagates through space from a transmitter to a receiver. The received signal strength depends on the euclidean distance d between the transmitter and the receiver and the characteristic of the environment the signal propagates in (e.g., indoor line of sight, indoor obstructed, outdoor free space, outdoor urban, etc.). The path loss model defines a parameter κ known as the path loss exponent, which is used to reflect the signal propagation characteristic of different environments. The power loss in the environment (expressed in dB) is predicted as $Loss = -10\kappa * \log_{10}(d)$. Some typical values for the path loss exponent in ns-2 are listed in [9].

The shadowing deviation model considers variations in the received signal strength for a given distance d. Power loss is calculated as $Loss = -10\kappa * \log_{10}(d) + sh\_dev$, where sh_dev is a Gaussian random variable with zero mean and standard deviation σ. Hence, increasing the value of σ means that the variation in the received signal strength will be higher for any given distance d.

In ns-2, a single value for each κ and σ is used in the network and a different value of sh_dev is generated for each packet that is received anywhere in the network. In reality, however, network conditions may be different at different locations and may fluctuate in epochs of time rather than with every packet. We modified the shadowing propagation model in ns-2 so that the value of sh_dev changes every t seconds (rather than with every packet), where t is generated according to an exponential distribution with mean of 5 seconds. When a node receives a packet, the received signal strength will depend on the value of κ and the current value of sh_dev. Hence, network conditions remain stable for a period of time.

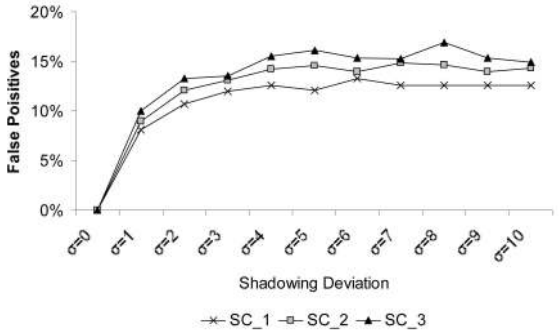

Fig. 16. False positives of the three simulation scenarios.

In our simulation, we used path loss exponent $\kappa = 2.5$ (which can model communication in outdoor urban environments and some indoor office environments [9], [10]). We increase σ from 0 to 10 to model different levels of fluctuation in network conditions, where $\sigma = 10$ indicates high variation in the received signal strength for a given distance d. This causes node behavior to vary by up to 20 percent (i.e., the packet forwarding ratio of a node may vary by up to 20 percent depending on the network condition).

Figs. 16 and 17 show the false positives and false negatives of our adaptive system. The total number of SPRT decisions made ranged between 3,000 and 7,000 (for the values of σ between 0 and 10). The highest false positives reported is 13.32 percent for SC-1, 14.81 percent for SC-2, and 16.86 percent for SC-3. All values are close to α as noted in Section 4.1. The highest false negatives reported are 8.4 percent for SC-2 and 9.1 percent for SC-3. SC-1 has no false negatives since no misbehavior exists in that case. All false negative values are bounded by β.

The results from both the static and the dynamic environments show the shortcomings of the nonadaptive reputation management systems. A nonadaptive reputation management system requires the value of TH to be carefully set and constantly updated to reflect the network

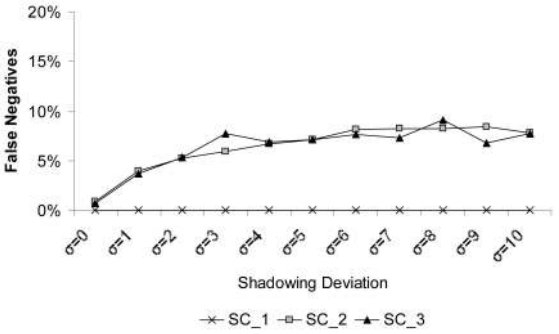

Fig. 17. False negatives of the three simulation scenarios.

conditions where it operates (otherwise, the system accuracy will be compromised). In contrast, results show that our adaptive reputation management system works well under both static and dynamic environments. It automatically self-adapts its criteria for identifying cooperative and misbehaving behaviors according to the network condition it is operating in.

5.3 System Overhead

In this section, we discuss the computation, communication, and storage overhead of our adaptive reputation management system on nodes in an ad hoc network.

Our adaptive reputation management system relies on localized observations to assess node behavior. Our system has no communication overhead since we do not rely on the exchange of observations between nodes. Exchange of observations may increase the system promptness but it also increases communication overhead and exposes the system to misreporting and collusion-based attacks (such as the slander attack [6]), which decreases system accuracy. In addition, network conditions may vary spatially. Hence, reflecting on observations about the behavior of distant nodes without knowing the network conditions under which these observations were made may reduce the accuracy of node behavior evaluation.

Our mechanism requires nodes to gather localized observations (RCV, FWD, and DRP events) that reflect their neighbors' behavior. This requires each node to act in promiscuous mode (capturing packets that may not be destined to it) and distinguish data packets that reflect RCV or FWD events related to its neighbors. Each node then stores packet traces for data packets that reflect an RCV event in a lookup table in order to identify any subsequent FWD events. Each RCV event in the lookup table will have a time-out associated with it. If an FWD event is not noted within the duration of that time-out, a DRP event is triggered and the corresponding RCV event entry in the lookup table is purged. This process results in computation and storage overhead.

In [1], we considered a number of approaches to reduce the types of computation and storage overheads that our adaptive reputation management system may induce. One approach we considered is hashing of packet trace entries. We evaluated this approach and showed that it reduces the size of the lookup table significantly and has almost no effect on the system effectiveness. A second technique that we considered is to set an upper bound on the size of the lookup table. We evaluated this approach as well and showed that it can reduce overhead.

While computation and storage overhead result from nodes acting in promiscuous mode (processing and storing traces of packets), we believe that this overhead is low but most importantly warranted especially when considering the impact of misbehavior on the network if a reputation management system is not implemented [1].

While mobility was not explicitly considered in this work, it is known that reputation-based mechanisms are most reliable when mobility is limited. Buchegger et al. [11] concluded that reputation mechanisms are most effective when based on direct observation. In [12], Buttyan and Hubaux note that node mobility can have a negative effect on such systems and suggest that mobility should be low enough to allow efforts carried during the evaluation and detection phases to be recouped in the reaction phase. An investigation of the performance of our adaptive reputation management system under varying levels of mobility is left for future work. We expect our system, similar to other systems, to perform better at lower levels of mobility. However, our adaptation capability should give our system the edge over others.

6 Related Work

Reputation management systems have been proposed to address insider misbehavior security concerns (i.e., selfish as well as malicious misbehavior) in self-organized communication systems such as ad hoc networks [7], [9], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], peer-to-peer systems [24], and in systems utilizing artificial intelligence [25]. Reputation management systems for ad hoc networks can be classified based on the source of the observations used by the evaluation function to assign reputation scores. The first class uses only observations that are collected locally by a node. These local observations used can be collected either passively or actively. Passively collected observations are usually acquired by promiscuously monitoring neighbors' actions. Actively collected observations are based on the receipt of evidence, such as an acknowledgment, which could be used as an indication of node behavior. An example of a system that uses passively collected observations is the watchdog mechanism introduced in [20]. For each packet that a node forwards, it monitors its next hop neighbor through which the packet was forwarded to ensure that it further forwards the packet to the next hop node toward the packet's destination. The drawbacks of the watchdog mechanism are analyzed in [23]. In [7], a testbed is used to evaluate the watchdog mechanisms under a number of misbehavior attacks. In contrast, reputation management systems proposed in [13], [17], [18], [21] rely on actively collected observations. These systems use acknowledgments received for each packet sent as indications of good behavior. The lack of acknowledgments or the presence of retransmissions is used as indications of misbehavior.

The second class of reputation mechanisms collects observations based on hearsay (the sharing of observations that are indicative of node behavior among nodes in the network) [16], [19], [22]. In [23], Yau and Mitchell qualitatively analyze the drawbacks and attacks against systems where observations are shared and indicate that the most reliable evaluations are those which a node makes based on local observations. Sharing observations has potential drawbacks, mainly related to overhead, misreporting, and collusion [23]. The overhead of sharing escalates as the size of the network increases. However, He et al. [19] succeed in limiting sharing to local neighbors, thus, reducing communication overhead. Moreover, nodes may falsely report observations that are indicative of bad behavior (to damage the reputation of others). In [14], the robustness of a reputation management system that employs sharing was increased against false accusations by introducing a Bayesian-based estimation mechanism. Additionally, Buchegger and Boudec [14] introduced a node redemption technique that readmits

misbehaved nodes to the network for reevaluation, and a reputation fading technique that mitigates against sudden exploitation of built-up reputation. On the other hand, systems that employ sharing of observations can detect node misbehavior more quickly compared to systems that rely on local observations [14], [15]. This is due to the increased amount of information regarding a particular node's behavior.

In [26], a sequential probability ratio test based algorithm was introduced to detect misbehavior at the MAC layer in ad hoc networks. In [8], we introduced a sequential probability ratio test based algorithm to detect selfish misbehavior with respect to packet forwarding in ad hoc networks. The algorithm relied on actively collected localized observations (while the work on this paper relies on passively collected localized observations) by requiring destination-based acknowledgments for each packet sent by the source.

EigenTrust [25] is a reputation management system for peer-to-peer networks. The EigenTrust algorithm is based on the notion of transitive trust. It suggests that if a peer i trusts another peer j, it can also trust all the peers that peer trusts. It also assumes a set of pretrusted peers, which are the ones a new peer joining the P2P network would trust. Trust relationships are built, thereafter, based on interactions between peers. EigenTrust addresses mostly the issues of collusion among malicious users but does not explicitly consider adaptation to varying peer behavior.

PeerTrust [26] is another reputation management system for peer-to-peer networks. It considers five trust parameters in evaluating trustworthiness of a peer. Two of these parameters are adaptive. One of the adaptive parameters considers the context of a transaction. This takes into consideration factors such as the time and size of a transaction in an attempt to profile the behavior of a peer as a function of these factors (e.g., does the peer act cooperatively when the size of a transaction is small but maliciously otherwise?). The notion of adaptation in PeerTrust is different from the notion of adaptation we consider in this work. PeerTrust adapts to varying peer behavior while our system adapts to dynamically changing network conditions. PeerTrust also uses the notion of personalized similarities to rate the credibility of peers. Credibility of peers is needed to assess the credibility of feedback given by a peer to another (this is not needed in our system since we rely solely on localized observations and do not share any trust/reputation information). To rate the credibility of a peer P2, a given peer P1 will use the similarities between itself and P2 to weight the feedback by P2 on other peers. Table 6 summarizes the similarities and differences between EigenTrust, PeerTrust, and this work.

To the best of our knowledge, no work prior to ours has proposed adaptation of the reputation management functions to dynamically changing network conditions in ad hoc networks.

TABLE 6
Illustrating Differences between EigenTrust, PeerTrust, and Our Adaptive RMS

EigenTrust	PeerTrust	Adaptive RMS
Operates in P2P networks	Operates in P2P networks	Operates in ad hoc networks
Application layer RMS	Application layer RMS	Network layer RMS
Evaluates humans	Evaluates humans	Evaluates nodes
Relies on feedback from peers (requires sharing of trust/reputation information)	Relies on feedback from peers (requires sharing of trust/reputation information)	Relies solely on local observations
	Adapts to varying peer behavior	Adapts to dynamically changing network conditions
	Personalized similarities are used to rate the credibility of peers	Personalized similarities are used to evaluate the behavior of nodes

7 Conclusion

In this paper, we have demonstrated the importance of adaptation in reputation management systems in dynamic network environments such as ad hoc networks. We introduced adaptive evaluation and detection functions, demonstrated their effectiveness in a dynamic ad hoc network, and compared the results against a nonadaptive system. The results obtained show that our adaptive system can operate under a wide range of network conditions and yields low false positives and false negatives with fast detection, thus, reducing the impact of misbehaving nodes on the network.

In our future work, we will assess the effectiveness of our adaptive reputation management system under more complex network environments where network conditions, network structure (e.g., topology, node density, etc.), and nodes' intent (i.e., to act cooperatively versus to misbehave) may change rapidly.

Acknowledgments

Part of this work was done while M.T. Refaei was at NIST, Gaithersburg, MD 20899.

References

[1] M.T. Refaei, "Adaptation in Reputation Management Systems for Ad hoc Networks," PhD dissertation, Virginia Polytechnic Inst. and State Univ., 2007.
[2] X. Yingqi and L. Wang-Chien, "Exploring Spatial Correlation for Link Quality Estimation in Wireless Sensor Networks," *Proc. Fourth Ann. IEEE Int'l Conf. Pervasive Computing and Comm.,* 2006.
[3] T.K. Forde, L.E. Doyle, and D. O'Mahony, "Ad Hoc Innovation: Distributed Decision Making in Ad Hoc Networks," *IEEE Comm. Magazine,* vol. 44, no. 4, pp. 131-137, Apr. 2006.
[4] A. Wald, *Sequential Analysis.* J. Wiley & Sons, 1947.
[5] Y.-C. Hu, A. Perrig, and D.B. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks," *Proc. MobiCom '02,* pp. 12-23, 2002.

[6] 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.
[7] S. Buchegger, C. Tissieres, and J.-Y.L. Boudec, "A Test-Bed for Misbehavior Detection in Mobile Ad-Hoc Networks—How Much Can Watchdogs Really Do?" *Proc. Sixth IEEE Workshop Mobile Computing Systems and Applications (WMCSA)*, pp. 102-111, 2004.
[8] M.T. Refaei, Y. Rong, L. DaSilva, and H.-A. Choi, "Detecting Node Misbehavior in Ad Hoc Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '07)*, pp. 3425-3430, 2007.
[9] J. Liu and V. Issarny, "Enhanced Reputation Mechanism for Mobile Ad Hoc Networks," *Proc. Second Int'l Conf. Trust Management (iTrust)*, 2004.
[10] K. Pahlavan, *Wireless Information Networks.* J. Wiley & Sons, 2005.
[11] S. Buchegger, J. Mundinger, and J.-Y.L. Boudec, "Reputation Systems for Self-Organized Networks: Lessons Learned," *IEEE Technology and Soc. Magazine*, vol. 27, no. 1, pp. 41-47, Mar. 2008.
[12] L. Buttyan and J.-P. Hubaux, *Security and Cooperation in Wireless Networks.* Cambridge Univ. Press, 2007.
[13] K. Balakrishnan, J. Deng, and P. Varshney, "TWOACK: Preventing Selfishness in Mobile Ad Hoc Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, vol. 4, pp. 2137-2142, 2005.
[14] S. Buchegger and J.-Y.L. Boudec, "A Robust Reputation System for Mobile Ad Hoc Networks," EPFL Technical Report No. IC/2003/50, July 2003.
[15] S. Buchegger and J.-Y.L. Boudec, "The Effect of Rumor Spreading in Reputation Systems for Mobile Ad-Hoc Networks," *Proc. Modeling and Optimization in Mobile Ad Hoc and Wireless Networks (WiOpt)*, Mar. 2003.
[16] S. Buchegger and J.Y. LeBoudec, "Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes—Fairness In Dynamic Ad-hoc NeTworks)," *Proc. ACM MobiHoc*, pp. 226-236, June 2002.
[17] M. Conti, E. Gregori, and G. Maselli, "Towards Reliable Forwarding for Ad Hoc Networks," *Proc. Personal Wireless Comm. (PWC)*, pp. 790-804, 2003.
[18] P. Dewan, P. Dasgupta, and A. Bhattacharya, "On Using Reputations in Ad Hoc Networks to Counter Malicious Nodes," *Proc. 10th Int'l Conf. Parallel and Distributed Systems (ICPADS)*, pp. 665-672, July 2004.
[19] Q. He, D. Wu, and P. Khosla, "SORI: A Secure and Objective Reputation-Based Incentive Scheme for Ad Hoc Networks," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, vol. 2, pp. 825-830, 2004.
[20] S. Marti, T.J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *Proc. MobiCom*, pp. 255-265, Apr. 2000.
[21] M.T. Refaei, V. Srivastava, L. DaSilva, and M. Eltoweissy, "A Reputation-Based Mechanism for Isolating Selfish Nodes in Ad Hoc Networks," *Proc. Second Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services (Mobiquitous)*, pp. 3-11, July 2005.
[22] P. Michiardi and R. Molva, "CORE: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," *Proc. IFIP TC6/TC11 Sixth Joint Working Conf. Comm. and Multimedia Security*, pp. 107-121, 2002.
[23] P.-W. Yau and C.J. Mitchell, "Reputation Methods for Routing Security for Mobile Ad Hoc Networks," *Proc. Joint First Workshop Mobile Future and Symp. Trends in Comm. (SympoTIC)*, pp. 130-137, Oct. 2003.
[24] K. Aberer, Z. Despotovic, W. Galuba, and W. Kellerer, "The Complex Facets of Reputation and Trust," *Computational Intelligence, Theory and Application*, Springer, 2006.
[25] J. Sabater and C. Sierra, "Review on Computational Trust and Reputation Models," *Artificial Intelligence Rev.*, vol. 24, pp. 33-60, 2005.
[26] Y. Rong, S.K. Lee, and H.-A. Choi, "Detecting Stations Cheating on Backoff Rules in 802.11 Networks Using Sequential Analysis," *Proc. 25th Joint Conf. IEEE Computer and Comm. Soc.*, 2006.

Mohamed Tamer Refaei received the PhD degree from Virginia Tech's Bradley Department of Electrical and Computer Engineering in 2007. He is currently a senior information security engineer/scientist at the MITRE Corporation in McLean, Virginia. Prior to working at the MITRE Corporation, he worked at the National Institute of Standard and Technologies. His research interests include reputation management systems for ad hoc networks, cognitive radio networks, and delay-tolerant networks.

Luiz A. DaSilva joined Trinity College Dublin in 2009, where he currently holds the Stokes Professorship in telecommunications in the Department of Electronic and Electrical Engineering. He has also been a faculty member at Virginia Tech's Bradley Department of Electrical and Computer Engineering since 1998. His research focuses on distributed and adaptive resource management in wireless networks, and in particular cognitive radio networks. He is a senior member of the IEEE and a member of the ASEE and of the ACM. In 2006, he was named a college of engineering faculty fellow at Virginia Tech.

Mohamed Eltoweissy received the MS and BS degrees in computer science and automatic control from Alexandria University, Egypt, in 1989 and 1986, respectively, and the PhD degree in computer science from Old Dominion University in 1993. He is an associate professor in the Bradley Department of Electrical and Computer Engineering at Virginia Tech. He also holds a courtesy appointment in the Department of Computer Science. His research interests are in the areas of information assurance and trust, networking and security in large-scale, ubiquitous cyber-physical systems, and group communications. His contributions include concern-oriented architecture and cell-based network model, dynamic key management in sensor networks, and elastic sensor-actuator networks. He has more than 130 publications in archival journals and respected books and conference proceedings. He is a senior member of the IEEE and a senior member of the ACM, the ACM SIGBED, and the ACM SIGSAC. He received the nomination for the Virginia SCHEV outstanding faculty awards, the highest honor for faculty in Virginia.

Tamer Nadeem received the PhD degree from the University of Maryland, College Park, in 2006. He is currently a research scientist at Siemens Corporate Research in Princeton. His research interest includes radio management for wireless networks, vehicular networking, pervasive computing, sensor networks, and peer-to-peer systems. He has published more than 40 technical papers in refereed conferences and journals including the *Journal on Selected Areas of Communication (JSAC)*, the *IEEE Transaction on Mobile Computing*, the IEEE Infocom, and the ACM IMC. He is a member of the ACM, the IEEE, the IEEE Computer Society, and the IEEE Communication Society.