MAC-Layer Selfish Misbehavior in IEEE 802.11 Ad Hoc Networks: Detection and Defense

Ming Li, *Member, IEEE*, Sergio Salinas, *Student Member, IEEE*, Pan Li, *Member, IEEE*, Jinyuan Sun, *Member, IEEE*, and Xiaoxia Huang, *Member, IEEE*

Abstract—In ad hoc networks, selfish nodes deviating from the standard MAC (Medium Access Control) protocol can significantly degrade normal nodes' performance and are usually difficult to detect. In this paper, we propose detection and defense schemes to identify and defend against MAC-layer selfish misbehavior, respectively, in IEEE 802.11 multi-hop ad hoc networks. Specifically, the non-deterministic nature of the IEEE 802.11 MAC protocol imposes great challenges to distinguishing selfish nodes from well-behaved nodes. Most traditional selfish misbehavior detection approaches are for wireless local area networks (WLANs) only. They either rely on a large amount of historical data to perform statistical detection, or employ throughput or delay models that are only valid in WLANs for detection. In contrast, we propose a realtime selfish misbehavior detection scheme for multi-hop ad hoc networks. It requires only several samples, and hence is more efficient and can adapt to channel dynamics more quickly. Then, based on the proposed detection scheme, we design three selfish misbehavior defense schemes against three typical kinds of smart selfish nodes. We find that the smart selfish nodes cannot degrade normal nodes' performance much without getting detected. Extensive simulation results are finally presented to validate the proposed detection and defense schemes.

Index Terms—Selfish misbehavior, IEEE 802.11 MAC, detection, defense

1 Introduction

Communication protocols are usually designed under the assumption that all participants would comply with the regulations. However, in untrusted communication environments, a misbehaving user can deviate from the regulations and cause damage to or obtain performance gain over other honest parties. Thus, trustworthy communication is a crucial issue, especially in wireless ad hoc networks where nodes need to fully cooperate with each other to ensure correct route establishment, successful packet delivery, and efficient resource usage. Traditional approaches to providing network security are mostly cryptography based. Unfortunately, they cannot be used to address user misbehavior at the Medium Access Control (MAC) layer.

MAC layer misbehavior can be generally classified into the following two categories: malicious misbehavior [1]-[4] and selfish misbehavior [5]-[7]. Malicious misbehavior primarily aims at disrupting the normal operations of a network. One kind of malicious misbehavior is jamming attack [1], [8], which is one particular type of Denial-of-Service (DoS) attack [2], [9], [10]. The malicious users could either constantly generate strong signals to overwhelm normal nodes' signals, or transmit fake packets to occupy the shared channel and hence prevent the normal users from communicating. Another kind of typical malicious misbehavior is Sybil attack [11], [12], where malicious users forge a large number of pseudonymous identities, and use them to substantially disturb or control the network. In contrast, selfish users can deliberately deviate from the standard MAC protocol to gain more network resources over well-behaved nodes. Generally, selfish nodes can benefit in the following two scenarios: first, obtaining a large portion of channel sharing, and second, reducing power consumption, e.g., by denying the forwarding of incoming packets. Compared to the second scenario, the first one is more difficult to detect, and can result in more serious problems and greatly degrade normal users' performances.

In this paper, we focus on the MAC layer selfish misbehavior problem in wireless ad hoc networks, in particular, IEEE 802.11 ad hoc networks, where selfish nodes aim to obtain higher MAC layer performance. We address this issue from two perspectives: *detection* and *defense*.

Selfish misbehavior detection. In IEEE 802.11 networks, selfish nodes can manipulate the following MAC layer parameters to enhance their channel access probabilities: the remaining transmission duration contained in frames, SIFS duration, DIFS duration, and backoff time. The most challenging detection task is to detect backoff time manipulation [13]. The difficulty primarily stems from the non-deterministic nature of IEEE 802.11 MAC that does not allow a straightforward way of distinguishing between a normal transmitter, which happens to select short backoff time, and a selfish node that intentionally selects short backoff time [10].

In the literature, there are a few works on the detection of backoff time manipulation. Radosavac et al. [5] propose a sequential probability ratio test (SPRT) algorithm to address the detection problem. However, SPRT is a parametric

- *M. Li is with the Department of Computer Science and Engineering, University of Nevada, Reno, NV 89557. E-mail: mingli@unr.edu.*
- *S. Salinas and P. Li are with the Department of Electrical and Computer Engineering, Mississippi State University, Mississippi State, MS 39762. E-mail: sas573@msstate.edu, li@ece.mmstate.edu.*
- *J. Sun is with the Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996. E-mail: jysun@eecsutk.edu.*
- *X. Huang is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China. E-mail: xx.huang@siat.ac.cn.*

Manuscript received 2 Sept. 2013; revised 25 May 2014; accepted 2 Aug. 2014. Date of publication 14 Aug. 2014; date of current version 1 May 2015. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TMC.2014.2348560

statistical approach which means that prior knowledge of the selfish nodes' behavior needs to be present at the detectors. Compared to [5], some other detection schemes, such as DOMINO [13], O-DOMINO [14], CUSUM [15], and [16], do not require any prior knowledge of selfish nodes. But all these schemes are designed for wireless local area networks (WLANs) only and cannot be directly applied to multi-hop ad hoc networks. The main reasons are as follows. First, some previous schemes [13]-[15] rely on a large amount of historical data to perform statistical detection. For example, CUSUM [15] requires up to 10^6 samples to determine whether a node is selfish or not. Compared with our detection scheme, which only requires several samples to make a decision, the detectors of CUSUM should be rich in storage and computation resources. Besides, since their schemes are based on the analysis over a large amount of data, the detection may take a very long time. Specifically, to collect 10^6 data transmission samples from one node in IEEE 802.11 wireless networks may take thousands of seconds or even more. These may not be affordable for normal nodes in ad hoc networks. Second, many schemes like [13]-[16] are based on throughput or delay models which are only valid in WLANs.

Noticing the above problems, we propose a realtime detection scheme for multihop ad hoc networks, which requires only several samples and no prior knowledge of selfish nodes. Specifically, instead of simply comparing the backoff time of a node under observation with the expected backoff time of normal nodes like in [13], [14], an observer compares the joint probability Y of the events that normal nodes would choose backoff times shorter than those in the chosen samples with the expectation of the joint probability $\mathbf{E}[Y]$ multiplied by a *detection factor* μ. If $Y \leq \mu\mathbf{E}[Y]$, the observer then classifies the node under observation as a selfish node. We also define another parameter called *confidence level* α to quantify the confidence of the detection results. Thus, we can set μ in the detection rule according to the expected confidence level. Moreover, since one selfish node is usually watched by multiple observers, they send their reports to a local cluster head who employs the majority rule to make the final decision. To complete our work, we also briefly discuss how to detect the manipulation of the other three MAC layer parameters as well.

In addition, the detection scheme is carried out periodically. Once a node is determined by a local cluster head as a selfish node, a penalty scheme is applied to punish the detected selfish node by decreasing its throughput. Specifically, all its one-hop neighbors will stop forwarding packets for it until they receive another decision notice indicating that this node is not selfish any more.

Selfish misbehavior defense. After detecting selfish nodes, we then need defend against such selfish misbehavior. A couple of papers exist in the literature addressing the defense against backoff time manipulation problem. Kyasanur and Vaidya [6] develop a scheme in which a detected selfish transmitter would be required by the corresponding receiver to use a longer backoff time, assuming that the receiver is a normal node. A similar approach is proposed in [17]. Konorski [18], [19] designs game theoretic approaches for WLANs which are resilient to selfish misbehavior. Besides, with modifications to the IEEE 802.11 binary exponential backoff (BEB) scheme, Guang et al. [20] propose to force each node to generate a predictable contention window (CW) size. Any node who picks a smaller backoff value (contention window size) than the predicted one would be regarded as a selfish node. Note that all the above approaches need greatly modify the standard IEEE 802.11 MAC protocol.

In this paper, based on the proposed detection scheme, we design three defense schemes against three typical kinds of selfish nodes manipulating their backoff times, i.e., naive selfish nodes, random selfish nodes, and γ-persistent selfish nodes. In particular, a naive selfish node always chooses a small constant value as its backoff time. A random selfish node randomly chooses its backoff time from a smaller fixed contention window than that of normal nodes. A γ-persistent selfish node still follows the IEEE 802.11 BEB rule to double its contention window size in case of retransmissions. However, the backoff time will be determined by multiplying a randomly chosen value in current contention window by a control parameter γ. Observers can tell which kind of selfish node a node would be by monitoring its transmissions. Besides, we consider these selfish nodes smart in the sense that they aim to gain more channel access under the condition that they do not get caught by the observers (due to the penalty scheme), whose primary objective is to prevent selfish nodes from causing damage to the network. Under the proposed defense scheme, we find that the selfish nodes cannot degrade normal nodes' performance much without getting detected.

The rest of this paper is organized as follows. In Section 2, we introduce IEEE 802.11 MAC protocol and the selfish misbehavior model discussed in this paper. We then present the selfish misbehavior detection and selfish misbehavior defense schemes in Sections 3 and 4, respectively. Simulations are conducted in Section 5 to evaluate the performance of the detection and defense schemes. We finally conclude this paper in Section 6.

2 Preliminaries

2.1 IEEE 802.11 MAC

IEEE 802.11 Distributed Coordination Function (DCF) uses Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) as the fundamental access scheme to resolve the contention among multiple nodes accessing the same channel [21]. In particular, before a node begins to transmit, it has to sense the channel to determine if there is any ongoing transmission. If the channel is busy, the node shall defer its transmission until the channel is sensed idle for a period of DIFS. After that, the node randomly chooses a backoff period in the contention window $[0, CW-1]$, where CW is maintained by each node, starts a backoff timer and counts down. The backoff timer decreases by 1 after the channel is sensed idle for the duration of a particular backoff slot. If the channel is sensed busy during any slot in the backoff interval, the backoff timer will be suspended. It can be resumed only after the channel is sensed idle for a period of DIFS again. After the backoff timer reduces to zero, the node may start its transmission by following a four-way handshake procedure (RTS/CTS/DATA/ACK). Specifically, the sender first sends an RTS to the receiver. After correctly receiving the RTS, the receiver responds with a CTS a period of SIFS later. Similarly, after correctly receiving the CTS, the sender begins to transmit the DATA a period of SIFS later. This transmission ends after the receiver correctly receives the DATA and responses with an ACK. In this process, all the four kinds of frames contain an estimated duration of the rest of the transmission. Other nodes that receive these frames update their Network Allocation Vector (NAV) with the duration, and are only allowed to transmit after they sense the channel idle for a period of DIFS after their NAVs expire. Finally, if the four-way handshake process is successful, the sender will reset its contention window size to a minimum value CW_{min}. Otherwise, if the transmission is unsuccessful (detected by the absence of a CTS or an ACK), the sender will double its contention window size, up to a maximum value CW_{max}.

2.2 Selfish Misbehavior Model

In a network, normal nodes follow network protocols to transmit or receive messages, while selfish nodes manipulate their local protocols in order to achieve higher performance (e.g., throughput). We classify selfish nodes into two types: *dumb selfish nodes* and *smart selfish nodes*. Dumb selfish nodes are aggressive and just aim to gain higher performance. Smart selfish nodes, however, are more cautious and intend to obtain better performance without getting busted by normal nodes. For instance, smart selfish nodes can learn normal nodes' detection schemes and change their own behavior adaptively. Comparing these two types of selfish nodes, we can easily see that dumb selfish nodes are much easier to detect than smart ones. In this study, we will first develop a general detection scheme for both kinds of selfish nodes and then propose some defense mechanisms to make sure that smart selfish nodes cannot affect normal nodes' performance without getting caught. Note that most previous detection schemes do not discuss what to do with the selfish nodes after detecting them. In this paper, we employ a simple penalty scheme to punish the detected selfish nodes, which is to let all their one-hop neighbors stop forwarding packets for them until they receive a notice indicating that these nodes are not selfish any more.

Besides, in this work we study MAC layer selfish misbehavior in IEEE 802.11 ad hoc networks. In such networks, selfish nodes can manipulate the following MAC layer parameters to enhance their channel access probability: duration of the rest of the transmission (or the remaining transmission duration), SIFS duration, DIFS duration, and backoff time. Specifically, when sending RTS or DATA frames, by increasing the included duration value, a selfish node can claim to occupy the channel for a longer period to prevent other normal nodes from contending for the channel. A selfish node may also choose a smaller SIFS duration so as to finish its current transmission sooner to initiate the next one. In addition, by setting DIFS to a smaller value after sensing the channel idle, a selfish node will wait a shorter time interval to start the backoff process and may have higher channel access probability. Moreover, when manipulating backoff time, selfish nodes may employ many different strategies in order to gain higher channel access probability. We consider three typical strategies herein as follows:

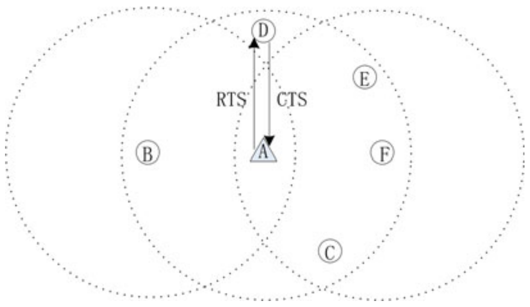


Fig. 1. Illustration of network topology.

- *Naive strategy*. A selfish node always chooses a small constant value as its backoff time.
- *Random strategy*. Instead of choosing a small constant backoff time, a selfish node randomly chooses its backoff time from a smaller fixed contention window than that of normal nodes, for example, $[0, CW_{min}/4]$. Thus, the selfish node's expected backoff period is smaller than that of normal nodes.
- *γ-Persistent strategy*. Instead of choosing a fixed contention window size, a selfish node still follows the IEEE BEB rule to double its contention window size in case of retransmissions. However, its backoff time is determined by multiplying a randomly chosen value in current contention window by a control parameter γ ($0 \leq \gamma \leq 1$), i.e., $T_B = t_b \cdot \gamma$ where T_B is the backoff time and t_b is the randomly chosen value in current contention window.

Note that the proposed detection scheme can be used to detect selfish nodes employing any strategies. We consider the above three typical selfish strategies when designing defense schemes against smart selfish nodes. The detection and defense schemes are carried out by observers, i.e., neighbors of the node of interest, and coordinated by local cluster heads who are known to be honest. For instance, the local cluster heads can be determined based on nodes' long-term behavior histories.

3 Selfish Misbehavior Detection

In this section, we propose to detect selfish misbehavior through normal nodes' observations. Specifically, we consider a multi-hop wireless network working on a single channel, in which every node is watched by all its neighbors. Normal nodes will compare the observed data with their counterparts under normal protocol operations, and apply the detection rules to determine whether the node under observation is a selfish node or not. Recall that in IEEE 802.11 networks, selfish nodes can manipulate four MAC layer parameters to gain higher channel access probability: the remaining transmission duration, SIFS duration, DIFS duration, and backoff time. We will address each of these issues in what follows.

3.1 Detection of Remaining Transmission Duration Manipulation

Fig. 1 shows a scenario where node A is the node of interest, while nodes B to F are all A's one-hop neighbors, i.e., observers. In the remaining transmission duration manipulation case, node A sets the duration contained in the RTS/

DATA frames it sends out to a larger value so that it can claim to occupy the channel for a longer period. Denote the duration contained in an RTS frame and the following DATA frame by D_{RTS} and D_{DATA}, respectively. When overhearing a DATA frame, nodes B to F can determine if A is a selfish node by checking the following two quantities:

$$T_R = D_{RTS} - 3 \times SIFS - T_{CTS} - T_{DATA} - T_{ACK}$$
$$T_D = D_{DATA} - SIFS - T_{ACK},$$

where T_{CTS}, T_{DATA}, and T_{ACK} are the duration of CTS, DATA, and ACK frames, respectively. T_{CTS} and T_{ACK} can be easily calculated based on the frame lengths specified in the IEEE 802.11 standard and the transmission rate, while T_{DATA} can be measured by the observing nodes. If $T_R > 0$ or $T_D > 0$, i.e., the duration contained in RTS or DATA frames is larger than the real duration of the rest of the transmission, then A is a selfish node.

3.2 Detection of SIFS Manipulation

In this case, selfish nodes choose a smaller SIFS duration after receiving a CTS so as to finish its current transmission sooner. Then, nodes B to F can infer A's SIFS duration as follows:

$$T_{SIFS} = t_{DATA} - t_{RTS} - T_{RTS} - SIFS - T_{CTS},$$

where t_{RTS} and t_{DATA} are the time instances at which an observer starts overhearing an RTS and the following DATA from A, respectively, and T_{RTS} is the duration of RTS frames. If $T_{SIFS} < SIFS$, then node A is a selfish node.

3.3 Detection of DIFS Duration/Backoff Time Manipulation

Both DIFS duration manipulation and backoff time manipulation intend to shorten the waiting period to initiate transmissions after the channel is sensed idle. Reducing the DIFS duration is equivalent to choosing a smaller backoff time. Therefore, we can detect these two cases using the same detection rules assuming the backoff time manipulation scenario. Since the backoff time is inherently a random variable, backoff time manipulation is much more difficult to detect compared to the previous two cases.

Before presenting the detection scheme, we first introduce how to estimate the observed node's backoff time and contention window size as follows.

- *Backoff time estimation*. Recall the operations of IEEE 802.11 DCF. If the channel is sensed busy during backoff process, the backoff timer will be suspended until the channel is sensed idle for a period of DIFS again. Thus, one node's transmission may be interleaved with one or more other nodes' transmissions. In order to accurately estimate the backoff time of a node under observation, we are interested in the time instances when the node sends out a DATA frame which is followed by an RTS frame from the same node. We still take Fig. 1 as an example. Denote the time instances at which node A starts transmitting a DATA frame and the following RTS frame by $t_{DATA,i-1}$ and $t_{RTS,i}$, respectively. During this period between $t_{DATA,i-1}$ and $t_{RTS,i}$, as an observer, a normal node may overhear (able to decode) or sense (unable to decode) other transmissions, i.e., some other nodes accessed the channel before A. The normal nodes decide that several frames on the channel are for the same transmission if the inter-frame period is equal to SIFS. Denote the number of such transmissions between $t_{DATA,i-1}$ and $t_{RTS,i}$ by K ($K \geq 0$) and the duration for each transmission by T_k where $0 \leq k \leq K$ and $T_0 = 0$. Then, node A's backoff time, denoted by BT_A, can be estimated by

$$\begin{aligned} BT_A = t_{RTS,i} - DIFS - (t_{DATA,i-1} + T_{DATA,i-1}) \\ - SIFS - T_{ACK} - \sum_{k=0}^{K} T_k - K \cdot DIFS \end{aligned} \tag{1}$$

where $T_{DATA,i-1}$ is the duration of the DATA frame transmitted by A at $t_{DATA,i-1}$.[1]
- *Contention window size estimation*. In order to find the contention window size for a transmitter, say node A, a neighboring node, say node B, can keep tracking the RTS frames A has transmitted. According to IEEE 802.11 MAC protocol, when a node transmits an RTS for a DATA frame for the first time, the contention window size starts from CW_{min}. If any collision occurs, the node will double its contention window size for its RTS retransmission. Therefore, node B only needs to count the number of times it overhears an RTS frame sent by A after A's last DATA transmission to infer A's contention window size assuming A is following the standard MAC protocol.[2]

Next, we propose a detection scheme to detect the selfish nodes manipulating their DIFS duration or backoff time to access the channel with higher priority. We mainly rely on the one-hop neighbors to perform the detection.

Note that a normal node randomly chooses its backoff time, denoted by BT, from a contention window $[0, CW - 1]$, where CW is its current contention window size. Thus, the probability that a node's BT is less than or equal to an estimated backoff time t is[3]:

$$\mathbf{P}[BT \leq t] = \frac{t+1}{CW},$$

where CW can be inferred according to the discussions above. Besides, the above probability itself can also be considered a random variable, which we denote by X, since t is essentially uniformly chosen in the contention window. Thus, if a node is well-behaved, the expectation of X should be

$$\mathbf{E}[X] = \sum_{t=0}^{CW-1} \frac{t+1}{CW} \cdot \frac{1}{CW} = \frac{CW+1}{2CW}.$$

1. Note that we assume selfish nodes always have packets to send so that they do not stay idle at the MAC layer.

2. Notice that in the case that a DATA frame transmission fails or no ACK is received, the DATA frame will be retransmitted. Since the DATA frame will be labeled according to IEEE 802.11 MAC, observers can know it is a retransmitted DATA frame and adjust their estimations accordingly.

3. Note that this probability is in fact a conditional probability. We just use the current notations for simplicity.

Since X is a random variable, it is very difficult to determine whether a node is a selfish node based on one single sample of X. Therefore, we extend the above expression to the case of multiple observed samples in the following.

Note that the backoff times of consecutive observed samples are correlated if they are for the same DATA frame (one DATA frame might be retransmitted several times with multiple backoff processes). Thus, an observer chooses multiple observation samples for different DATA frames which are thus uncorrelated and independent. In particular, consider n such chosen observation samples. Denote the detected backoff time and the corresponding contention window size for the ith ($1 \leq i \leq n$) sample by t_i and CW_i, respectively, and

$$X_i = \mathbf{P}[BT_i \leq t_i] = \frac{t_i + 1}{CW_i}.$$

Then, $X_1, \ldots, X_n$ are independent of each other. We can have their joint cumulative distribution function (CDF), which we denote by Y, as

$$\begin{aligned} Y &= \mathbf{P}[BT_1 \leq t_1, \ldots BT_n \leq t_n] \\ &= \mathbf{P}[BT_1 \leq t_1] \cdot \cdots \cdot \mathbf{P}[BT_n \leq t_n] \\ &= \frac{t_1 + 1}{CW_1} \cdot \cdots \cdot \frac{t_n + 1}{CW_n}, \end{aligned}$$

and the expectation as

$$\mathbf{E}[Y] = \mathbf{E}\left[\frac{t_1 + 1}{CW_1}\right] \cdot \cdots \cdot \mathbf{E}\left[\frac{t_n + 1}{CW_n}\right] = \prod_{i=1}^{n} \frac{CW_i + 1}{2CW_i}. \quad (2)$$

Thus, if an observer detects that

$$Y \leq \mu \mathbf{E}[Y], \quad (3)$$

it then considers the node under observation as a selfish node, where μ ($0 < \mu \leq 1$) is called *the detection factor*, a control parameter. In the detection rule, different values of μ will lead to different detection probabilities. How to set μ depends on the design goal we want to achieve: if we want to identify more selfish nodes in the networks, a larger μ is needed; on the other hand, if we allow a few selfish nodes to exist and do not want the well-behaved nodes to be wrongly classified as selfish nodes, a smaller μ is more appropriate.

Furthermore, we introduce another parameter called *confidence level* to quantify the confidence of the detection results given μ, which is denoted by α and defined as follows:

$$\alpha = 1 - \mathbf{P}\big[Y \leq \mu \mathbf{E}[Y]\big] = \mathbf{P}\big[Y > \mu \mathbf{E}[Y]\big]. \quad (4)$$

Notice that $\mathbf{P}\big[Y \leq \mu \mathbf{E}[Y]\big]$ is the probability that a normal node is classified as a selfish node. For example, suppose according to the detection rule, we find that $Y \leq \mu \mathbf{E}[Y]$ and hence classify the node under observation as a selfish node. Assume $\mathbf{P}\big[Y \leq \mu \mathbf{E}[Y]\big] = 0.05$. Then it means that if this node is a normal node, the event that $Y \leq \mu \mathbf{E}[Y]$ occurs with a probability of 0.05. Thus, we say that our confidence level is 0.95 to classify the node as a selfish node. While in another case, assume $\mathbf{P}\big[Y \leq \mu \mathbf{E}[Y]\big] = 0.95$. Then if the node under observation is a normal node, the event that $Y \leq \mu \mathbf{E}[Y]$ occurs with a probability of 0.95. Therefore, we are much less confident to classify the node as a selfish node, or with a confidence level of only 0.05.

Clearly, confidence level α has a close relationship with the detection factor μ: a larger μ leads to a smaller value of α, and a smaller μ results in a larger α. However, calculating α is a non-trivial problem, and we present how to efficiently calculate α in the following. Specifically, we take $\ln(\cdot)$ on the both sides of the inequality $Y > \mu \mathbf{E}[Y]$ in (4), which leads to

$$\alpha = \mathbf{P}\left[\ln\left(\frac{t_1 + 1}{CW_1}\right) + \cdots + \ln\left(\frac{t_n + 1}{CW_n}\right) > \ln\big(\mu \mathbf{E}[Y]\big)\right].$$

Let $Z_i = \ln \frac{t_i + 1}{CW_i}$ where $1 \leq i \leq n$. Then Z_i is a random variable with probability density function (PDF): $p_{Z_i}(z_i) = \frac{1}{CW_i}$ where $z_i \in \left\{ \ln \frac{j}{CW_i} | 1 \leq j \leq CW_i \right\}$. Consequently, the confidence level α can be rewritten as

$$\alpha = \mathbf{P}\{Z_1 + \cdots + Z_n > \theta\}$$

where $\theta = \ln(\mu \mathbf{E}[Y])$. Since the observation samples are uncorrelated, $Z_1, \ldots, Z_n$ are independent random variables.

As we know, the PDF of the sum of two independent random variables is the convolution of their individual PDFs. This property can be easily extended to the sum of a number of independent random variables. Let $Z = Z_1 + \cdots + Z_n$, and $p_Z(z)$ be the PDF of Z. Then, we can have

$$p_Z(z) = \bigotimes_{i \in \{1, \ldots, n\}} p_{Z_i}(z_i)$$

where $\otimes$ denote the convolution operator. In order to efficiently calculate the linear convolution, we can first zero-pad all the sequences, and compute the Discrete Fourier Transform (DFT) of each sequence using the fast Fourier transform (FFT) algorithm (e.g., Cooley-Tukey algorithm). After that, we point-by-point multiply the DFTs of all the sequences, where the product represents the DFT of the PDF convolution,[4] i.e.,

$$\mathscr{F}\{p_Z(z)\} = \prod_{i \in \{1, \ldots, n\}} \mathscr{F}\{p_{Z_i}(z_i)\}.$$

Finally, we compute the inverse DFT (IDFT) of $\mathscr{F}\{p_Z(z)\}$ to obtain $p_Z(z)$.

At last, we can obtain that

$$\alpha = \mathbf{P}[Z > \theta] = \sum_{\{z | z > \theta\}} p_Z(z),$$

which depends on CW_i ($1 \leq i \leq n$) and μ. Thus, when we design the detection rules, we should also take the confidence level into consideration. For instance, given a lower bound on the confidence level, we then have an upper

4. The purpose of using the fast Fourier transform (FFT) to compute the DTF is to reduce the computational complexity when calculating the convolution among the PDFs of Z_i's. According to the results in [22], the computational complexity of direct convolution between two vectors, each with size of K, is $O(K^2)$. After applying FFT, the computational complexity to obtain the convolution becomes $O(K \log K)$, which is much less than $O(K^2)$. Therefore, when calculating the convolution among n sequences, the computational complexity can be greatly reduced after applying FFT.

bound on the detection factor μ. Besides, usually one selfish node is watched by several observers. Each observer will employ the above detection scheme to perform detection and send their decisions to a local cluster head (e.g., by piggybacking), which then employs the majority rule to make the final decision.

Note that the proposed scheme needs only several data samples and no prior knowledge of selfish nodes. Thus, it is more efficient and takes less time than previous schemes [13]-[15], which rely on a large amount of historical data to perform statistical detection. Moreover, the proposed detection is carried out periodically. Once a node is determined by a local cluster head as a selfish node, a penalty scheme is applied to punish the detected selfish node by decreasing its throughput for a certain time period. Specifically, all its one-hop neighbors will stop forwarding packets for it until they receive another decision notice indicating that this node is not selfish any more. In other words, each detection result is effective until a new detection result is obtained.

Moreover, our scheme discussed above deals with a single selfish node and does not include the scenario of colluding nodes. Recall that observers will send their detection results to a local cluster head, which then employs the majority rule to make the final decision. Therefore, as long as there are enough honest nodes (or more honest nodes than selfish nodes), selfish nodes can still be detected even though they collude with each other.

3.4 Hidden Terminal Problems

In addition, we notice that the hidden terminal problem may affect the performance of the proposed detection scheme. Hidden terminal problem is a common problem in multi-hop wireless networks. Many schemes have been proposed to address this problem, e.g., by using a out-of-band busy tone [23], [24] or tuning physical carrier sensing ranges [25], [26]. In fact, many previous detection schemes proposed for WLANs, such as [13]-[16], are also affected by the hidden terminal problem since some nodes may not hear each other. In general, hidden terminals can potentially result in the following two misdiagnosis cases in our scheme. Let us take Fig. 1 as an example. Assuming that B cannot hear E, C, F which have ongoing transmissions, E, C, F are hidden nodes to B. In the first case of hidden terminal problems (HT-I) where A is observing B, as B is unaware of E's, C's, and F's transmissions, it continues to count down its backoff timer while A has suspended its. Then, B appears to A as a selfish node as it has a shorter backoff time than it really has. In the second case of hidden terminal problems (HT-II) where B is observing A, A suspends its backoff timer once it hears the transmissions from E, C, F. As B cannot hear these transmissions, A would have a longer backoff time than it really has according to (1). Consequently, if A is a selfish node, it is possible that A appears to B as a normal node.

For HT-I problems, recall that observers will send their detection results to a local cluster head, which then employs the majority rule to make the final decision in our scheme. When multiple observers' reports are jointly considered, even though some observers may have misdiagnosis, the final misdiagnosis probability could be greatly reduced. Similarly, For HT-II problems, misdiagnosis probability can also be mitigated by applying the majority rule. For example, if E, C, F can hear A, and thus make correct detections, then A can still be correctly identified if it misbehaves. Moreover, due to the above reason, a selfish node does not know whether it will finally be classified as being selfish or not, and hence cannot leverage the hidden node problem HT-II to increase its access probability. In addition, hidden terminal problems only happen to a limited number of nodes that are within the sensing range of one node and outside the sensing range of the other. HT-I and HT-II problems also mitigate each other. Therefore, as will be shown in Section 5, the confidential level and the detection probability obtained when both HT-I and HT-II problems exist are not very much different from those when there are no HT-I and HT-II problems.

4 Defense Schemes against Smart Selfish Nodes

As mentioned in Section 2, selfish nodes may employ three typical strategies: naive strategy, random strategy, and γ-persistent strategy to manipulate their backoff time, which we call naive selfish nodes, random selfish nodes, and γ-persistent selfish nodes, respectively. In this section, we propose defense schemes to prevent such selfish nodes from degrading normal nodes' performances. Recall that according to the penalty scheme, the throughput of a node will be reduced significantly if it is detected as being selfish. Thus, we consider "smart" selfish nodes here in the sense that they are afraid of being detected due to the penalty scheme. Note that the proposed defense schemes can also provide guidelines for defending other types of smart selfish nodes.

4.1 Naive Selfish Nodes and the Defense Scheme

Naive selfish nodes intend to choose a safe constant backoff time t_s so that they may gain higher channel access probability compared to normal nodes without getting busted by the observers, i.e.,

$$\frac{t_s+1}{CW_1}\cdot\cdots\cdot\frac{t_s+1}{CW_n} > \mu\mathbf{E}[Y].$$

Since an observer's detection scheme is closely related to its detection factor μ, a naive selfish node would have to adaptively change its transmission parameter based on the detection factors μ's chosen at all its observers in order not to be detected; otherwise, all its one-hop neighbors would stop forwarding its packets once it is caught. In this way, our defense schemes can effectually prevent naive selfish nodes from deviating from the regulations by having observers to reveal μ's through broadcasting. Therefore, revealing observers' detection factors is an important component of our defense schemes. Besides, to have selfish nodes know observers' detection factors can be easily realized by allowing each observer to broadcast its μ to its one-hop neighbors. This is sufficient to have each node be aware of all its observers' μ's because its observers are all its one-hop neighbors. Then, based on observers' detection rule in (3), which are also broadcasted to one-hop neighbors, μ's, and $\mathbf{E}[Y]$ shown in (2), naive selfish nodes can choose t_s as follows:

$$\begin{aligned} t_s &> \sqrt[n]{\left(\prod_{i=1}^{n} CW_i\right) \cdot \mu E[Y]} - 1 \\ &= \sqrt[n]{\left(\prod_{i=1}^{n} CW_i\right) \cdot \mu \prod_{i=1}^{n} \frac{CW_i + 1}{2CW_i}} - 1 \\ &= \frac{1}{2} \sqrt[n]{\mu \prod_{i=0}^{n} (CW_i + 1)} - 1. \end{aligned} \tag{5}$$

Notice that the CW_i in (5) may range from CW_{min} to CW_{max}. Since selfish nodes cannot know which samples the observers will choose and hence what value CW_i will have, they need to choose the safest value for CW_i, i.e., $CW_i = CW_{max}$, in (5) so that they will not be detected. Besides, each selfish node is usually observed by multiple observers with different μ's. Since the majority rule is applied to make the final decision based on multiple observers' reports, a safe constant backoff time can be chosen by

$$t_s = \left\lfloor \frac{1}{2}(CW_{max} + 1) \sqrt[n]{\mathrm{med}_{k \in N}\{\mu_k\}} \right\rfloor, \tag{6}$$

where N is the set of observers within one-hop of the selfish node, μ_k is the detection factor of the kth observer, $\lfloor x \rfloor$ is the largest integer that is no larger than x, and $\mathrm{med}\{\cdot\}$ is the median function, i.e., given a set of numbers $Q = \{q_1, q_2, \ldots, q_m\}$ sorted in ascending order, $\mathrm{med}\{Q\}$ returns the $\lceil \frac{m+1}{2} \rceil$th number.

On the other hand, from an observer's perspective, its main concern is to prevent the selfish nodes from degrading normal nodes' performance. Towards this end, how to choose a reasonable detection factor μ is a critical issue. From (6), we can see that a large μ can force the selfish nodes to choose a large t_s, and hence reduce their channel access probability. But, as shown in (4), a large μ would also result in degraded confidential level α of the detection results. Therefore, the defense strategy here is to choose the smallest μ which can make selfish nodes' chosen constant backoff time, i.e., t_s in (6), equal to or larger than the average of normal nodes' backoff time which we denote by $\mathbf{E}[t]$, i.e.,

$$\left\lfloor \frac{1}{2}(CW_{max} + 1) \sqrt[n]{\mathrm{med}_{k \in N}\{\mu_k\}} \right\rfloor \geq \mathbf{E}[t].$$

Thus, an observer, say k, will choose the following detection factor in its detection rule to make sure the above inequality holds:

$$\mu_k = \left(\frac{2\mathbf{E}[t]}{CW_{max} + 1}\right)^n. \tag{7}$$

However, there is still one question left regarding setting μ_k in (7): how should an observer calculate $\mathbf{E}[t]$? Since in multihop IEEE 802.11 networks, the physical carrier sensing range is larger than (usually around twice) the transmission range, selfish nodes' transmissions affect both their one-hop and two-hop neighbors. Thus, at each observer, $\mathbf{E}[t]$ is calculated as the average of its one-hop neighbors' backoff times t_j's, excluding the node under observation's, i.e., $\mathbf{E}[t] = \sum_{j \in S} t_j / |S|$, where S is the set of the current observer's one-hop neighbors excluding the node under observation, and $|S|$ means the number of nodes in S. Notice that every normal node's backoff time is a random number and depends on the local traffic conditions (since the contention window size does). So every normal node's instant backoff time may change dramatically which makes the calculation of $\mathbf{E}[t]$ and hence μ_k inaccurate. On the other hand, a long-term average of t_j will make $\mathbf{E}[t]$ insensitive to the changes in the communication environment. Thus, to account for the time-varying wireless environment as well as nodes' dynamic traffic loads, we employ the following adaptive filtering algorithm using the least mean-square (LMS) mechanism [27], [28] to update μ_k after a certain interval Δ

$$\mu_k(t) = \mu_k(t-1) + \omega \cdot \left[\frac{\overline{\tau}}{t_s} - \rho_{std}\right] \tag{8}$$

for $t \geq 1$, where $\mu_k(0)$ is set based on (7), $\mu_k(t)$ and $\mu_k(t-1)$ are the current and last detection factors, respectively, $\overline{\tau} = \sum_{j \in S} \overline{t}_j / |S|$ with $\overline{t}_j$ being the observed average backoff time of neighboring node j during the past time interval Δ, ρ_{std} is the ratio of $\overline{\tau}/t_s$ that we want to achieve, i.e., 1, and ω is the step size of this adaptive function. The observers update their detection factors after each updating interval Δ, and the selfish node changes its safe backoff time t_s accordingly each time. In addition, note that when the step size ω is large, the above adaptive filtering algorithm can adapt to the environment rapidly and is suitable for very dynamic channel conditions; while a small ω makes μ_k change slowly and is more suitable for wireless environments with mild channel dynamics.

4.2 Random Selfish Nodes and the Defense Scheme

Recall that random selfish nodes randomly choose a backoff time from a small constant contention window so as to make their expected backoff time smaller than that of normal nodes. Thus, different from naive selfish nodes which try to find the minimum safe backoff time t_s, random selfish nodes aim to determine the minimum safe contention window size CW_s so that they will not get caught by the observers.

Consider the n samples chosen by an observer. Let

$$Y' = \frac{t'_1 + 1}{CW_1} \cdot \cdots \cdot \frac{t'_n + 1}{CW_n},$$

where $t'_i \in \mathrm{U}[0, CW_s - 1]$ for $1 \leq i \leq n$. Thus, a random selfish node needs to choose its contention window size so that $\mathbf{E}[Y'] > \mu\mathbf{E}[Y]$, where $\mathbf{E}[Y]$ is shown in (2) and similarly,

$$\mathbf{E}[Y'] = \mathbf{E}\left[\frac{t'_1 + 1}{CW_1} \cdot \cdots \cdot \frac{t'_n + 1}{CW_n}\right] = \prod_{i=1}^{n} \frac{CW_s + 1}{2CW_i}.$$

Therefore, the random selfish node chooses constant contention window size as follows:

$$CW_s > \sqrt[n]{\left(\prod_{i=1}^{n} 2CW_i\right) \cdot \mu \prod_{i=1}^{n} \frac{CW_i + 1}{2CW_i}} - 1$$
$$= \sqrt[n]{\mu \prod_{i=1}^{n} (CW_i + 1)} - 1,$$

and similar to (6), a safe value of CW_s is

$$CW_s = \left\lfloor (CW_{max} + 1) \sqrt[n]{\text{med}_{k \in N}\{\mu_k\}} \right\rfloor. \tag{9}$$

On the other hand, along the line in Section 4.1, observers' major concern is how to choose μ to prevent the selfish nodes from severely degrading network performance. Therefore, the defense strategy here is to choose the smallest μ which can make selfish nodes' chosen constant contention window size, i.e., CW_s in (9), equal to or larger than the expectation of normal nodes' contention window size which we denote by $\mathbf{E}[CW]$, i.e.,

$$\left\lfloor (CW_{max} + 1) \sqrt[n]{\text{med}_{k \in N}\{\mu_k\}} \right\rfloor \geq \mathbf{E}[CW].$$

Thus, an observer k will choose the following detection factor in its detection rule to make sure the above inequality holds

$$\mu_k = \left(\frac{\mathbf{E}[CW]}{CW_{max} + 1}\right)^n. \tag{10}$$

We also employ a similar adaptive filtering algorithm to that shown in (8) to recursively update the detection factor μ_k at each observer

$$\mu_k(t) = \mu_k(t-1) + \omega \cdot \left[\frac{\overline{CW}}{CW_s} - \rho_{std}\right],$$

where $\overline{CW} = \sum_{j \in S} \overline{CW}_j / |S|$ with $\overline{CW}_j$ being the average contention window size of neighboring node j during the past time interval, ρ_{std} is the ratio of $\overline{CW}/CW_s$ that we want to maintain, i.e., 1, and $\mu_k(0)$ is set according to (10).

4.3 γ-Persistent Selfish Nodes and the Defense Strategy

As mentioned before, instead of choosing a constant contention window size, a γ-persistent selfish node still follows the IEEE 802.11 BEB rules to double its contention window size in case of retransmissions. However, the backoff time will be determined by multiplying a randomly chosen value in current contention window by a control parameter γ $(0 \leq \gamma \leq 1)$. γ-persistent selfish nodes' objective is to choose an appropriate γ so that they can get higher channel access probability without getting caught.

Again, consider the n samples chosen by an observer. Let

$$Y'' = \frac{\gamma t_1'' + 1}{CW_1} \cdot \dots \cdot \frac{\gamma t_n'' + 1}{CW_n},$$

where $t_i'' \in \text{U}[0, CW_i - 1]$ for $1 \leq i \leq n$. Thus, a γ-persistent selfish node needs to choose γ so that $\mathbf{E}[Y''] > \mu\mathbf{E}[Y]$, where $\mathbf{E}[Y]$ is shown in (2) and similarly,

$$\mathbf{E}[Y''] = \mathbf{E}\left[\frac{\gamma t_1'' + 1}{CW_1} \cdot \dots \cdot \frac{\gamma t_n'' + 1}{CW_n}\right] = \prod_{i=1}^{n} \frac{\gamma(CW_i - 1) + 2}{2CW_i}.$$

Then, to make $\mathbf{E}[Y''] > \mu\mathbf{E}[Y]$ is to let

$$\prod_{i=1}^{n} \frac{\gamma(CW_i - 1) + 2}{2CW_i} > \mu \prod_{i=1}^{n} \frac{CW_i + 1}{2CW_i}.$$

For the above inequality to hold, it is sufficient to have

$$\prod_{i=1}^{n} \gamma(CW_i - 1) \geq \mu \prod_{i=1}^{n} (CW_i + 1),$$

i.e., $\gamma \geq \mu \prod_{i=1}^{n} \frac{CW_i + 1}{CW_i - 1}$. Since CW_i's are unpredictable, a safe γ, denoted by γ_s, can be chosen as follows:

$$\gamma_s = \text{med}_{k \in N}\{\mu_k\} \left(\frac{CW_{min} + 1}{CW_{min} - 1}\right)^n.$$

In this case, an observer k's best strategy is to make the γ-persistent selfish node's expected backoff time no less than that of normal nodes, or $\gamma_s \geq 1$. To that end, it is sufficient to let

$$\mu_k \left(\frac{CW_{min} + 1}{CW_{min} - 1}\right)^n \geq 1,$$

which results in the following detection factor

$$\mu_k \geq \left(\frac{CW_{min} - 1}{CW_{min} + 1}\right)^n.$$

We also use the same adaptive filtering algorithm as that shown in (8) to recursively update the detection factor μ_k.

4.4 Selfish Misbehavior Classification

For the defense scheme, since smart selfish nodes may employ three typical strategies, an observer needs to determine which strategy a smart selfish node employs. As a smart selfish node's transmission parameter is closely related to the detection factors μ's of its observers, each observer can fix its μ when classifying a selfish node by analyzing its transmission parameter. Specifically, for a naive selfish node, its backoff time will be the same during multiple continuous samples if all its observers broadcast constant μ's according to (6). Therefore, a naive selfish node can be easily identified with several samples. For a random selfish node, when μ's are fixed, each observer can calculate the contention window size CWs of this node following (9). As a random selfish node uniformly chooses its backoff time from $[0, CW_s - 1]$, by applying statistical tests, e.g. Anderson-Darling test [29], observers can determine whether a set of backoff time samples satisfy uniform distribution over $[0, CW_s - 1]$. In particular, it only takes around 200 samples for an observer to reach the decision when applying Anderson-Darling test. For γ-persistent selfish nodes, we can follow the similar approach to identify them. According to the analysis above, if a selfish node adopts the same strategy, the selfish node classification process only needs to be conducted once. If a selfish node adaptively change its strategy, say, the change takes place before its

TABLE 1
Simulation Parameters

Parameters	Value
Channel frequency	2.4 GHz
Basic rate	1 Mbps
Data rate	2 Mbps
Transmission range	250 meters
Carrier sensing range	550 meters
(CW_{min}, CW_{max})	(32, 1,024)
RTS retry limit	7

observers determine which strategy it adopts in one round of classification, we propose to have its observers choose a stricter defense policy, i.e., to choose μ's by considering that this selfish node is a γ-persistent selfish node. (By comparing μ's obtained at the observers when a selfish node takes the three different strategies, we can easily know μ for γ-persistent selfish node is the largest, leading to the strictest detection policy among the three.) Thus, under this policy, a selfish node who changes its strategies frequently will not get a better performance, i.e., might have even higher detection probability, than it sticks to one strategy. In summary, the smart selfish node classification process only needs to run once in our defense scheme, which does not bring much overhead. For example, given that the length of DIFS, SIFS, a backoff time slot is 40, 10, and 20 us, respectively, the payload size of a data packet is 512 bytes, the data transmission rate is 2 Mbps, selfish misbehavior classification can be done in less than 1 second.

5 Simulation Results

In this section, we conduct extensive simulations in NS-2 (version 2.33) to evaluate the performance of the proposed detection, defense, as well as penalty schemes for IEEE 802.11 networks. Our network settings are as follows. We consider a network area of $1,000 \times 1,000\ m^2$ with some randomly distributed nodes (the number of nodes varies in different scenarios). We use the following shadowing channel model to simulate the variations in channel conditions over time and space:

$$\left[\frac{P_r(d)}{P_r(d_0)}\right]_{dB} = -10\beta log\left(\frac{d}{d_0}\right) + X_{dB},$$

where β is the path loss exponent, d is the distance between the transmitter and the receiver, $P_r(d_0)$ is the power at some reference distance d_0, and X_{dB} is a Gaussian random variable with zero mean and standard deviation σ_{dB}. We set β to 2 and σ_{dB} to 1 in the simulations. Besides, each node is a source for a flow and its corresponding destination is randomly chosen. Each flow has a Constant Bit Rate (CBR) traffic with data rate 500 Kbps and packet size 512 bytes. In addition, we adopt the basic and common settings at the MAC layer in NS-2. Some typical parameters are shown in Table 1.

Before presenting simulation results, we first define several evaluation metrics in the following:

- *Confidence level* (α). As defined in Section 3, confidence level is the probability that a normal node is classified as a well-behaved node.

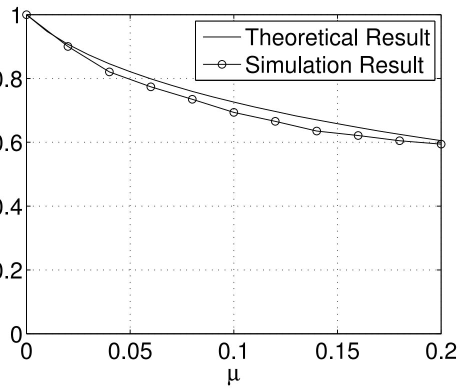


Fig. 2. The relationship between μ and α.

- *Detection probability*. This is defined as the probability that a selfish node is detected as a selfish node.
- *Channel occupation duration* and *channel occupation ratio* (r). These two parameters are used to evaluate nodes' channel usages. Specifically, we use normalized value of channel occupation duration, and calculate channel occupation ratio as one node's channel occupation duration divided by the average of its neighboring nodes' channel occupation durations.
- *Detection ratio*. Note that one selfish node is observed by multiple observers. The detection ratio is defined as the number of observers classifying a node as a selfish node divided by the total number of observers. Recall that a majority rule is employed by the local cluster head to finally determine whether a node under observation is selfish or not. In other words, a node would be classified as a selfish node if its detection ratio is larger than 0.5.

5.1 Selfish Misbehavior Detection

Here, we carry out simulations in a network of 20 nodes to verify the performance of selfish misbehavior detection schemes. Since the detection of remaining transmission duration manipulation and of SIFS manipulation are comparatively easier, we focus on the detection of DIFS duration/backoff time manipulations.

Confidence level. We first demonstrate the relationship between the confidence level α and the detection factor μ. During the simulation, the observed node works normally according to the IEEE MAC protocol. Fig. 2 shows both the theoretical results and the simulation results when the detection factor μ takes various values. We can see that α drops from 0.95 to around 0.6 while μ increases from 0.01 to 0.2. This shows that in order to obtain higher confidence level α, observers need to adopt a relatively small detection factor μ. For example, to have a confidence level α no less than 0.9, μ should be set to be no bigger than 0.02. Besides, we can find that the simulation results match theoretical results well.

In addition, we show in Fig. 3 the confidence levels under the four scenarios: with HT-I and HT-II, with HT-I without HT-II, without HT-I with HT-II, and without HT-I or HT-II. When there are only HT-I but not HT-II problems, the confidential level α is a little bit lower than that obtained when there are no HT-I or HT-II problems. It means when there

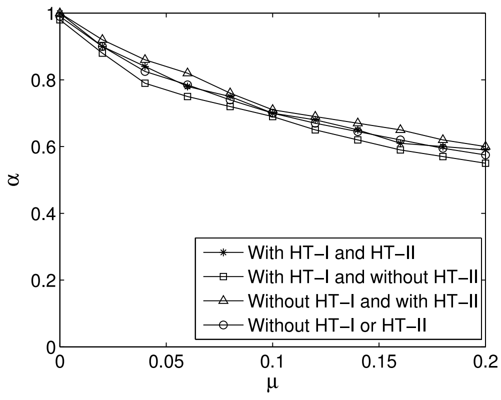


Fig. 3. α obtained with or without hidden terminal (HT) problems.

are HT-I problems, some normal nodes might be determined as selfish nodes. On the other hand, when there are only HT-II but not HT-I problems, the confidential level α is a little bit higher than that obtained when there are no HT-I or HT-II problems. It means when there are HT-II problems, more normal nodes would be determined as well-behaved ones. Interestingly, we find that the confidence level α obtained when both HT-I and HT-II problems exist is not very much different from that when there are no HT-I or HT-II problems. This is because the impact of both hidden terminal problems can mitigate each other on the performance of confidential level.

Channel occupation duration. We then study the channel occupation durations of selfish nodes and of normal nodes. Simulation results when selfish nodes are naive, random, and γ-persistent are shown in Figs. 4a, 4b, 4c, respectively. In Fig. 4a, as the constant backoff value adopted by naive selfish nodes grows, selfish nodes' average channel occupation duration drops while normal nodes' average channel occupation duration slightly increases. In particular, when selfish nodes set their backoff time to be 24 time slots, selfish nodes and normal nodes have about the same channel occupation durations, which means the selfish nodes do not affect normal nodes' performance in this case. Similar results can be observed in Figs. 4b and 4c, where selfish nodes and normal nodes have the same channel occupation duration when $CW = 45$ and $\gamma = 0.96$, respectively. Besides, we denote the percentage of selfish nodes among all the nodes as ρ, which we call "selfish node density". When $\rho = 50\%$, i.e., when $20 \times 50\% = 10$ nodes are randomly selected as selfish nodes, the channel occupation durations of selfish nodes and of normal nodes are lower than those of selfish nodes and of normal nodes, respectively, when $\rho = 30\%$ and when $\rho = 10\%$. The reason is that the selfish nodes will compete with each other, besides with normal nodes, to access the channel. The more selfish nodes there are, the lower their average channel occupation duration would be.

Detection probability. Next, we explore the detection probabilities when the detection factor μ is equal to 0.02, 0.05, and 0.1, and present the results in Fig. 5. Here, the selfish node density ρ is equal to 50 percent, which means that half of the nodes are selfish. In Fig. 5a, we can see that the detection probability decreases as the constant backoff time chosen by naive selfish nodes increases, which is consistent with our intuition. We also notice that the detection probability increases as μ increases. This is because when μ is larger, the detection rule is stricter. Thus, the cutoff backoff time beyond which the detection probability goes to 0 also increases as μ increases. Figs. 5b and 5c show similar results. Taking the case when $\mu = 0.05$ as an example, under the naive selfish model, detection probability is equal to 1 when the chosen constant backoff time is less than 12. However, under the random selfish model, when the chosen constant contention window size is 25 with expected backoff time of 12, the detection probability is less than 1. The detection probability is even lower under the γ-persistent selfish model when the control parameter $\gamma \leq 25/32 \approx 0.78$. Thus, it means that γ-persistent selfish nodes are smarter and more difficult to be detected than random selfish nodes, which are in turn smarter and more difficult to be detected than naive selfish nodes.

Besides, we show in Fig. 6 the detection probabilities under the following four scenarios: with HT-I and HT-II, with HT-I without HT-II, without HT-I with HT-II, and without HT-I or HT-II. We find that when there are only HT-I problems but not HT-II problems, the detection probability is a little bit higher than that obtained when there are no HT-I or HT-II problems. It means when there are HT-I problems, more selfish nodes would be found. On the other hand, when there are only HT-II problems but not HT-I problems, the detection probability is a little bit lower than that obtained when there are no HT-I or HT-II problems. It means when there are HT-II problems, some selfish nodes would be determined as normal nodes. Furthermore, we also find that the detection probability when both HT-I and HT-II problems exist is not very much different from that when there are no HT-I or HT-II problems due to the similar reason to that for the confidence level shown in Fig. 3.

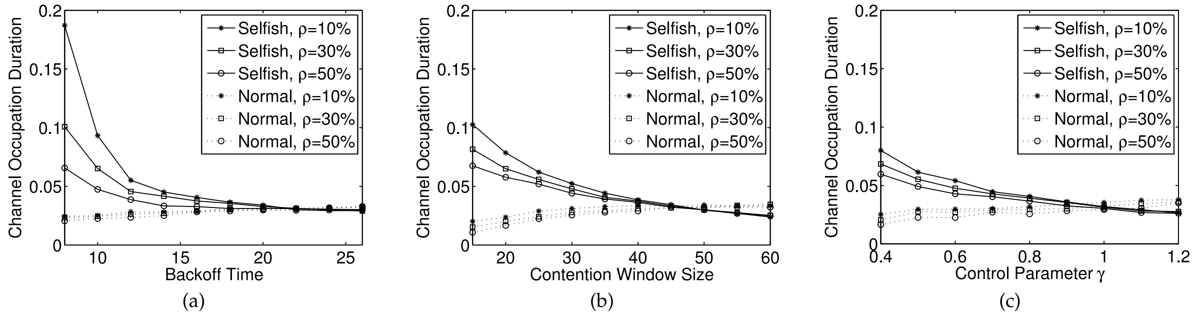


Fig. 4. Average channel occupation duration when $\rho = 10, 30,$ and 50 percent. a) Naive selfish nodes. b) Random selfish nodes. c) γ-persistent selfish nodes.

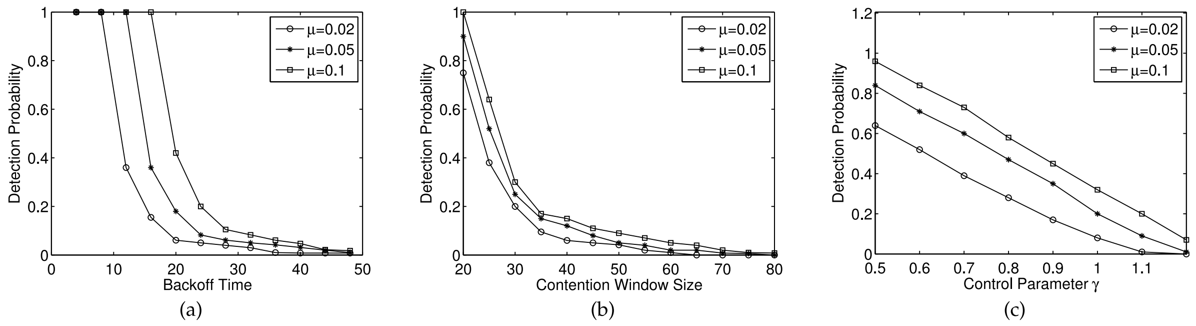


Fig. 5. Detection probabilities of three typical kinds of selfish nodes under different detection factors μ's. (a) Naive selfish nodes. (b) Random selfish nodes. (c) γ-persistent selfish nodes.

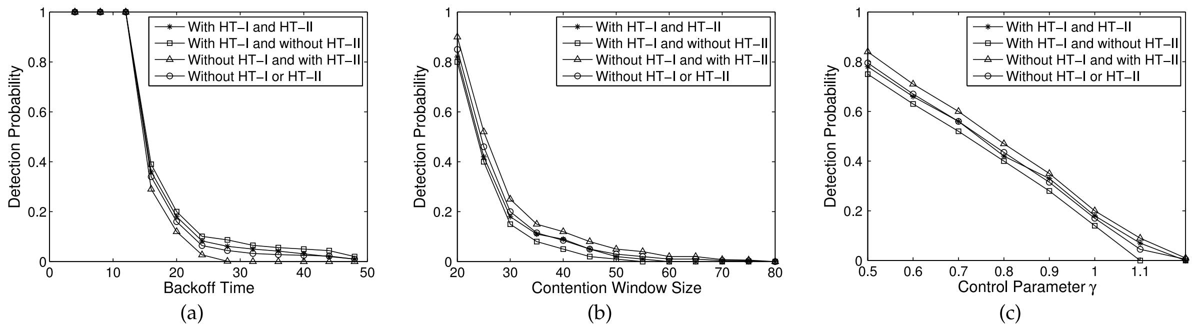


Fig. 6. Detection probability of three typical kinds of selfish nodes with and without hidden terminal (HT) problems. (a) Naive selfish nodes. (b) Random selfish nodes. (c) γ-persistent selfish nodes.

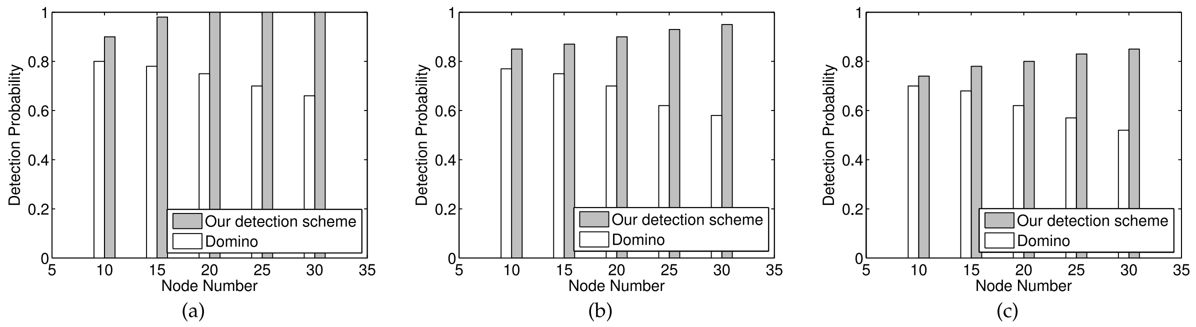
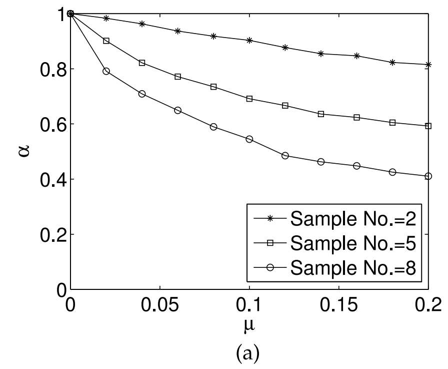


Fig. 7. Detection probability comparison of our scheme with DOMINO [15] for three typical kinds of selfish nodes. (a) Naive selfish nodes. (b) Random selfish nodes. (c) γ-persistent selfish nodes.

Next, we compare in Fig. 7 our scheme with DOMINO [15] when selfish nodes adopt different strategies as the number of nodes varies from 10 to 30. We find that our detection scheme can detect the selfish nodes with much higher detection probability compared with DOMINO in wireless ad hoc networks. Specifically, when there are 30 nodes, our detection scheme can achieve a detection probability of 1, 0.97, and 0.82 when selfish nodes take naive strategy, random strategy, and γ-persistent strategy, respectively; while DOMINO can only achieve a detection probability of 0.67, 0.58, and 0.53, respectively. This is because DOMINO is based on comparing one node's backoff time with the expected value in a WLAN, which is derived based on Bianchi's classic model [30] for WLANs.

Detection efficiency. Recall that our proposed detection scheme only requires a few samples. In what follows, we study the impact of the number of samples, i.e., n, on the detection performance. Due to page limit, we only give the simulation results for naive selfish nodes in Fig. 8. The results for the other two kinds of selfish nodes are similar. Specifically, as shown in Fig. 8a, when $n = 2$, in order to achieve $\alpha \geq 0.9$, observers should set their $\mu \leq 0.12$. At the same time, we can see from Fig. 8b that the detection probability is less than 0.35. Consequently, $n = 2$ may not be appropriate in this network when we need high detection probability. In the case of $n = 5$, Fig. 8 shows that $\alpha \geq 0.9$ is achieved when $\mu \leq 0.02$, with which the achievable detection probability can be up to around 0.6 as shown in Fig. 8b. Besides, when $n = 8$, $\alpha \geq 0.9$ can be achieved when $\mu \leq 0.01$, while at the same time the detection probability can be up to around 0.7. Comparing the results when $n = 5$ with those when $n = 8$, we notice that the latter case can have better performance when we have strict requirement on the confidence level. On the other hand, if we have high requirement on the detection probability, say no less than 0.9, the detection factor μ needs to be no less than 0.07 when $n = 5$ and no less than 0.02 when $n = 8$ as shown in Fig. 8b. Then, according to Fig. 8, we can find that under such conditions the confidence level is up to 0.75 when $n = 5$ and up to 0.8 when $n = 8$, which means that the latter case achieves better performance.



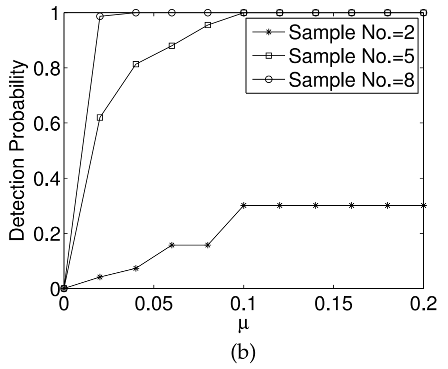


Fig. 8. Detection efficiency under different μ's and n's.

5.2 Selfish Misbehavior Defense

Recall that different from dumb selfish nodes, smart selfish nodes intend to choose safe misbehavior parameters to avoid being caught by the observers. Our defense scheme is, in general, to prevent the smart selfish nodes from affecting normal nodes' performance. In this section, we conduct simulations to evaluate the performance of the proposed defense scheme against the three typical types of smart selfish nodes.

Defense results with different network sizes. We first study the performance of our defense scheme with different network sizes. In particular, we consider sparse, semi-sparse, and dense networks with 10, 20, and 50 randomly distributed nodes, respectively, when selfish node density ρ is equal to 30 percent, and show the channel occupation ratio r in Figs. 9a, 9b, 9c for a randomly chosen naive, random, and γ-persistent selfish node, respectively. In the simulations, we choose $\Delta = 0.02\,s$ and $\omega = 10^{-2}$ as the interval and step size of the adaptive filtering algorithm in (8), respectively. We find that our defense scheme can make selfish nodes' channel occupation ratios around 1, i.e., they cannot affect normal nodes' performance. We also notice that as the number of nodes in the network gets larger, the variance of r becomes larger. The reason is that when the network gets more crowded, the communication environment and the channel condition become more dynamic. It hence becomes more difficult to accurately estimate some of the parameters in the defense scheme, e.g., $E[t]$, and $E[CW]$. Thus, we need reduce the interval Δ in the adaptive filtering algorithm (8) to enhance the performance of the defense scheme.

In addition, we show the detection ratios of our defense schemes in Figs. 9d, 9e, 9f for naive, random, and γ-persistent selfish nodes, respectively. Note that one selfish node is observed by multiple observers and that the detection ratio is defined as the number of observers classifying the selfish node as a selfish node divided by the total number of observers. Thus, if the detection ratio is above 0.5, it means that the observed node will be finally classified as a selfish node according to the majority rule mentioned in Section 3. From the simulation results, we find that the detection ratio is 0 for most of time, i.e., we cannot detect the selfish nodes, and is larger when the selfish node gains channel occupation ratio higher than 1. In other words, the selfish nodes cannot cause much performance degradation in the network without being detected.

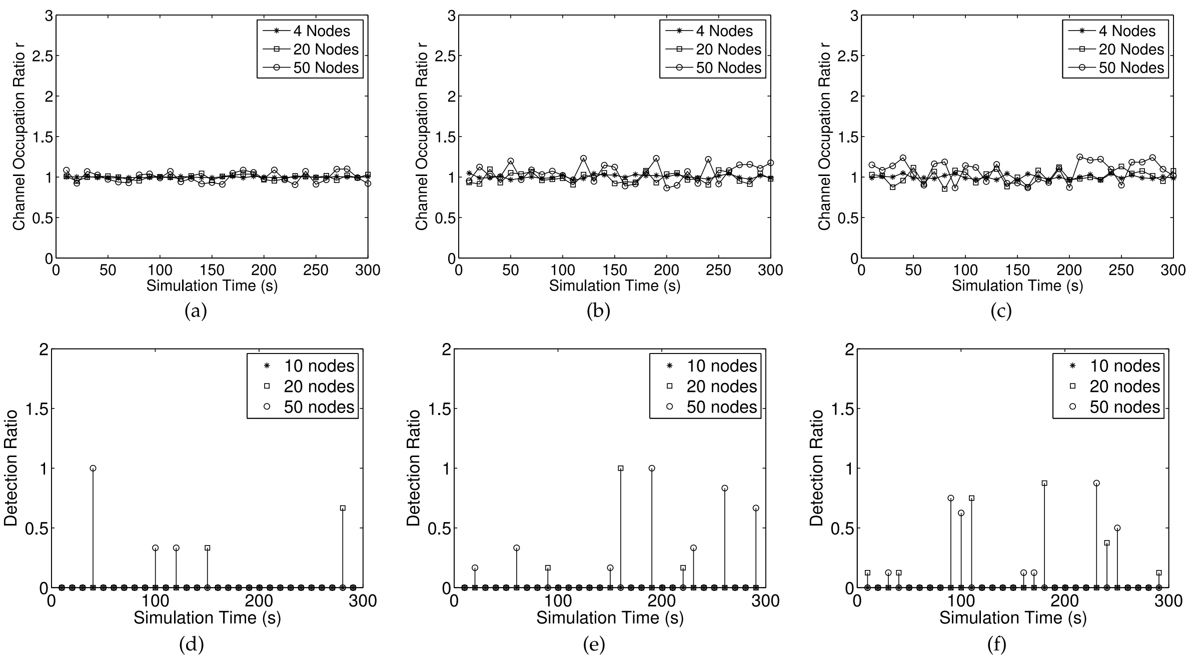
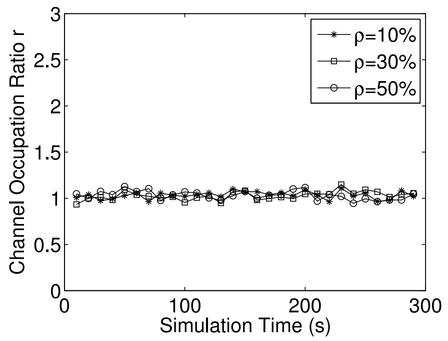
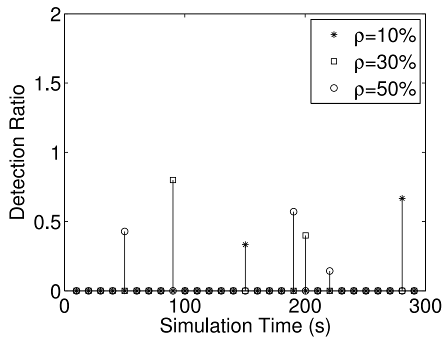


Fig. 9. Defense results with different network sizes. (a)-(c) Channel occupation ratios of naive selfish node, random selfish model, and γ-persistent selfish node, respectively. (d)-(f) Detection ratios of naive selfish node, random selfish model, and γ-persistent selfish node, respectively.



(a)



(b)

Fig. 10. Defense results under different selfish densities.

Defense results with different selfish node densities. Next, we verify the performance of our defense scheme against the naive selfish nodes with different selfish node densities in a network of 20 nodes. The results for the other two types of selfish nodes are similar and omitted due to page limit. We show the channel occupation ratio r when the selfish node density ρ is equal to 10, 30, and 50 percent in Fig. 10a. We can also find that our defense scheme makes selfish nodes' channel occupation ratios around 1. We also notice that the variances of r in all the three scenarios are almost the same. It indicates that our defense scheme can work well in networks with different selfish densities. It also explains why the number of times the selfish node is detected under these three scenarios are the same, which are all equal to 1, as shown in Fig. 10b.

5.3 Selfish Misbehavior Punishment

Then, we study the impact of our selfish misbehavior penalty scheme on both selfish nodes and normal nodes. We consider a network of 20 nodes with selfish node density ρ equal to 30 percent. Besides, we consider both dumb selfish nodes and smart selfish nodes. Due to page limit, we only present the results for naive selfish nodes here. As mentioned before, once a node is determined as a selfish node, all its one-hop neighbors will stop forwarding packets for it until a different decision is made. In our simulations, we let cluster heads collect detection results from observers and feed back the decision results to the observers every 5 seconds.

Fig. 11a shows the throughput of a randomly selected dumb selfish node, the throughput of one of its one-hop normal nodes, along with that of one of the other normal nodes. We can see that at the beginning, the dumb selfish node obtains around 3.5 times as much throughput as its neighboring node. Then, there follows a sharp decrease in its throughput, which finally goes to 0. The reason is that the dumb selfish node is very aggressive and just aims to gain more channel resource, which makes it easy to be detected. The dumb selfish node's throughput remains 0 since the detection is conducted periodically. We also notice that the dumb selfish node causes severe throughput degradation of its neighboring normal node since it still keeps trying to occupy the channel even though it has been detected. However, the impact is limited to its neighboring nodes only.

Moreover, Fig. 11b shows the throughput of a randomly selected smart selfish node, the throughput of one of its one-hop normal nodes, and that of one of the other normal nodes. We can observe that the throughput of the three nodes are nearly the same most of the time. Notice that the throughput of the smart selfish node has two sharp drops. We can infer that the selfish node is detected as being selfish twice during the simulation period. The reason why the throughput does not decrease to 0 is that some other nodes in the network have buffered some packets for the selfish node and can still forward these packets to the destinations. The results in Fig. 11 show that the penalty scheme can work well which will in turn motivate the smart selfish nodes to avoid being detected.

In addition, Fig. 11c shows the throughput of a randomly chosen normal node. We find that the normal node's throughput has two sharp drops, indicating that it has been misjudged as a selfish node twice during the entire simulation time. However, the penalty only lasts a few seconds each time. This is because once the cluster head finds the node as a normal node and sends this decision notice to its observers, they start forwarding packets for it again.

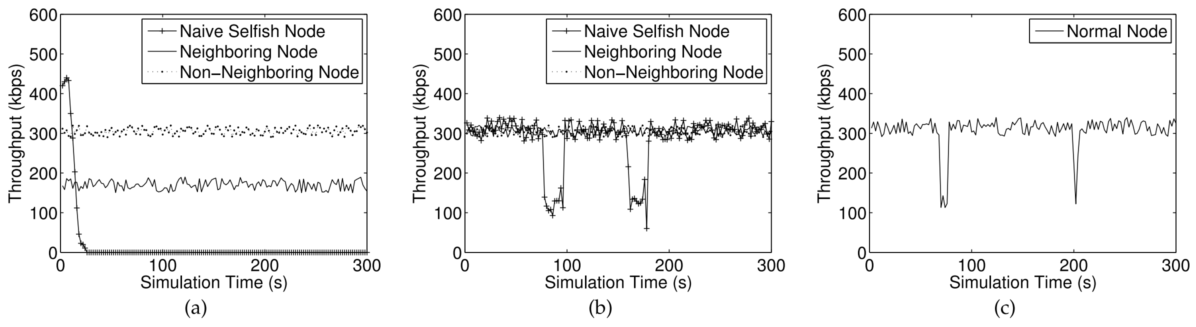


Fig. 11. Illustration of the proposed penalty scheme. (a) Dumb selfish nodes. (b) Smart selfish nodes. (c) Normal node.

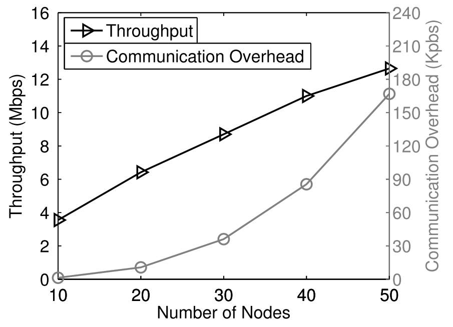


Fig. 12. Comparison of network throughput and scheme communication overhead.

5.4 Communication Overhead

We further illustrate the communication overhead incurred by our proposed schemes in this section. Specifically, in our selfish misbehavior detection scheme, the communication overhead stems from observers' uploading their detection results and the IDs of the observed nodes as well as their own to a local cluster head, and the cluster head's sending back its final decision.[5] In our selfish misbehavior defense scheme, the overhead includes the observers' broadcasting their detection factors μ_k's to their two-hop neighbors.[6] Fig. 12 compares the total node throughput and the total communication overhead of our scheme under different network sizes. We find that the total throughput of all nodes is 3.55, 6.42, 8.75, 10.97 and 12.61 Mbps, while the corresponding total communication overhead is 1.41, 10.71, 36.09, 85.48 and 166.89 Kbps, when the numbers of nodes in the network are 10, 20, 30, 40 and 50, respectively. Thus, the ratio of total communication overhead to total throughput is 1.41 Kbps/3.55 Mbps $= 0.40 \times 10^{-3}$, 10.71 Kbps/6.42 Mbps $= 1.67 \times 10^{-3}$, 36.09 Kbps/8.75 Mbps $= 4.12 \times 10^{-3}$, 85.48 Kbps/10.97 Mbps $= 7.79 \times 10^{-3}$, 166.89 Kbps/12.61 Mbps $= 1.32 \times 10^{-2}$, when the numbers of nodes in the network are 10, 20, 30, 40 and 50, respectively. Obviously, the total overhead incurred by our schemes is negligible compared to total network throughput, i.e., the proposed schemes have negligible impact on network performance.

6 Conclusions

In wireless ad hoc networks, selfish nodes that deliberately deviate from the standard MAC protocol may obtain an unfair share of the channel resource and degrade the performance of other well-behaved nodes. In this paper, we have presented a distributed observation based selfish misbehavior detection scheme, which has low computation complexity and can quickly adapt to channel dynamics. We have also developed three defense schemes against three typical kinds of selfish nodes. These schemes can ensure that the selfish nodes cannot degrade normal nodes' performance much without getting detected. Finally, we conduct extensive simulations and the results show that the proposed schemes can work well.

5. We use 1 byte, 4 bytes and 1 byte to record a detection result, a node's ID and a feedback decision, respectively.

6. We use 1 byte to record each μ_k.

Acknowledgments

The work of P. Li was partially supported by the U.S. National Science Foundation under grants CNS-1149786 (CAREER Award), ECCS-1128768, CNS-1343220, and CNS-1147851. The work of Huang was supported by the National High Technology Research and Development Program of China (863 program) under grant No. 2011AA010503, and the Joint Program of National Science Foundation of China-Guangdong under grant No. U1301256.

References

[1] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proc. 6th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Urbana-Champaign, IL, USA, May 2005, pp. 46–57.

[2] A. L. Toledo, and X. Wang, "Robust detection of MAC layer denial-of-service attacks in CSMA/CA wireless networks," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 3, pp. 347–358, Sep. 2008.

[3] M. Manzo, T. Roosta, and S. Sastry, "Time synchronization attacks in sensor networks," in *Proc. ACM 3rd Workshop Security Ad Hoc Sensor Netw.*, Alexandria, VA, USA, Nov. 2005, pp. 107–116.

[4] D. R. Raymond, R. C. Marchany, and M. I. Brownfield, "Effects of denial-of-sleep attacks on wireless sensor network MAC protocols," *IEEE Trans. Veh. Technol.*, vol. 58, no. 1, pp. 367–380, Jan. 2009.

[5] S. Radosavac, J. S. Babaras, and L. Koutsopoulos, "A framework for MAC protocol misbehavior detection in wireless networks," in *Proc. 4th ACM Workshop Wireless Security*, Cologne, Germany, Sep. 2005, pp. 33–42.

[6] P. Kyasanur and N. Vaidya, "Selfish MAC layer misbehavior in wireless network," *IEEE Trans. Mobile Comput.*, vol. 4, no. 5, pp. 502–516, Sep. 2005.

[7] Z. Lu, W. Wang, and C. Wang, "On order gain of backoff misbehaving nodes in CSMA/CA-based wireless networks," in *Proc. IEEE Conf. Comput. Commun.*, San Diego, CA, USA, Mar. 2010, pp. 1–9.

[8] Y. Law and M. Palaniswami, "Energy-efficient link-layer jamming attacks against wireless sensor network MAC protocols," *ACM Trans. Sens. Netw.*, vol. 5, no. 1, pp. 347–358, Feb. 2009.

[9] K. El-Khatib, "Impact of feature reduction on the efficiency of wireless intrusion detection systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 21, no. 8, pp. 1143–1149, Aug. 2010.

[10] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of service attacks at the MAC layer in wireless ad hoc networks," in *Proc. IEEE Mil. Commun. Conf.*, Anaheim, CA, USA, Oct. 2002, pp. 1118–1123.

[11] T. Zhou, R. R. Choudhury, and P. Ning, "P2dap-sybil attacks detection in vehicular ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 3, pp. 582–594, Mar. 2011.

[12] H. Yu, P. B. Gibbons, and M. Kaminsky, "Sybillimit: A near-optimal social network defense against sybil attacks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 885–898, Jun. 2010.

[13] M. Raya, J. Hubaux, and I. Aad, "Domino: A system to detect greedy behavior in ieee 802.11 hotspots," in *Proc. ACM 2nd Int. Conf. Mobile Syst., Appl. Serv.*, Boston, MA, USA, Jun. 2004, pp. 84–97.

[14] A. A. Cardenas, S. Radosavac, and J. S. Baras, "Performance comparison of detection schemes for MAC layer misbehavior," in *Proc. IEEE Conf. Comput. Commun.*, Anchorage, Alaska, May 2007, pp. 1496–1504.

[15] J. Tang, Y. Cheng, and W. Zhuang, "An analytical approach to real-time misbehavior detection in IEEE 802.11 based wireless networks," in *Proc. IEEE Conf. Comput. Commun.*, Shanghai, China, Apr. 2011, pp. 1638–1646.

[16] N. Jaggi, V. R. Giri, and V. Namboodiri, "Distributed reaction mechanisms to prevent selfish misbehavior in wireless ad hoc networks," in *Proc. IEEE Global Telecommun. Conf.*, Houston, TX, USA, Dec. 2011, pp. 1–6.

[17] A. A. Cardenas, S. Radosavac, and J. S. Baras, "Detection and prevention of MAC layer misbehavior in ad hoc networks," in *Proc. ACM 2nd Workshop Security Ad Hoc Sensor Netw.*, Washington, DC, USA, Oct. 2004, pp. 17–22.

[18] J. Konorski, "Protection of fairness for multimedia traffic streams in a non-cooperative wireless LAN setting," in *Proc. 6th Int. Conf. Protocols Multimedia Syst.*, 2001, pp. 116–129.

[19] J. Konorski, "Multiple access in ad-hoc wireless lans with noncooperative stations," in *Proc. 2nd Netw. Conf.*, 2002, pp. 1141–1146.
[20] L. Guang, C. M. Assi, and A. Benslimane, "Enhancing IEEE 802.11 random backoff in selfish environment," *IEEE Trans. Veh. Technol.*, vol. 57, no. 3, pp. 1806–1822, May 2008.
[21] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE802.11, 1999.
[22] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Math. Comput.*, vol. 19, pp. 297–301, 1965.
[23] Z. J. Haas and J. Deng, "Dual busy tone multiple access (DBTMA)—A multiple access control scheme for ad hoc networks," *IEEE Trans. Commun.*, vol. 50, no. 6, pp. 975–985, Jun. 2002.
[24] A. C. V. Gummalla and J. O. Limb, "Wireless collision detect (WCD): Multiple access with receiver initiated feedback and carrier detect signal," in *Proc. IEEE Int. Conf. Commun.*, New Orleans, LA, USA, Jun. 2000, pp. 397–401.
[25] H. Zhai and Y. Fang, "Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks," in *Proc. IEEE Conf. Comput. Commun.*, Barcelona, Spain, Apr. 2006, pp. 1–12.
[26] I. W.-H. Ho, and S. C. Liew, "Impact of power control on performance of IEEE 802.11 wireless networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 11, pp. 1245–1258, Nov. 2007.
[27] B. Widrow and S. D. Steams, *Adaptive Signal Processing*, Englewood Cliffs, NJ, USA: Prentice-Hall, 1985.
[28] R. H. Kwong, and E. W. Johnston, "A variable step size LMS algorithm," *IEEE Trans. Signal Process.*, vol. 40, no. 7, pp. 1633–1642, Jul. 1992.
[29] T. W. Anderson and D. A. Darling, "Asymptotic theory of certain 'Goodness of Fit' criteria based on stochastic processes," *Ann. Math. Stat.*, vol. 23, no. 2, pp. 193–212, 1952.
[30] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535–547, Mar. 2000.