A Lightweight Passive Online Detection Method for Pinpointing Misbehavior in WLANs

Jaehyuk Choi, *Member*, *IEEE*, Alexander W. Min, *Member*, *IEEE*, and Kang G. Shin, *Fellow*, *IEEE*

Abstract—Detecting misbehaving users in wireless networks is an important problem that has been drawing considerable attention. Even though there is a plethora of work on 802.11 wireless local area networks (WLANs), most existing schemes employ behavior-based anomaly detection, assuming that the backoff-time information of each transmitting node is available to the monitoring node. Unfortunately, it is practically infeasible to obtain the accurate backoff value chosen by other transmitting nodes because this MAC-layer information is not readily available. In this paper, we propose a practical way of pinpointing the misbehaving nodes without requiring access of hardware-level (e.g., backoff time) information in 802.11 WLANs. In contrast to most prior work, our scheme exploits the sequence of successfully received packets, which are readily observable at the access point. The distinct features of our scheme are that it 1) promptly detects a misbehaving node using a sequential hypothesis test, 2) performs well in realistic erroneous channel conditions due to its ability to accurately capture link heterogeneity, and 3) incurs negligible memory and computation overheads as it makes detection decisions based on runtime observations. The effectiveness of the proposed scheme is evaluated via extensive simulation as well as implementation, demonstrating its capability of accurately detecting nodess' selfish behavior in realistic 802.11 WLAN environments.

Index Terms—Network monitoring, IEEE 802.11, WLANs, passive online detection, driver-level solution, greedy behavior.

1 Introduction

Recent advances in radio technology, such as Software-Defined Radios (SDRs) [1], [2], open-source drivers [3], and reverse-engineered firmware [4], allow users to modify their wireless interface software and change the protocol parameters to meet their own needs. This programmability provides flexibility to end users to best suit their performance needs, such as connectivity and quality of service (QoS) [5]. However, a misbehaving user can abuse this flexibility to increase his own throughput by manipulating the channel access functions in a selfish manner, at the cost of other well-behaving users' performance. This selfish problem can pose a serious threat to network performance and fairness. Therefore, it is important to detect such misbehaving users and mitigate their impact on the performance of other well-behaving users.

In IEEE 802.11 wireless local area networks (WLANs), the selfish users commonly achieve their greed by manipulating the MAC parameters associated with channel access, such as contention window (CW) size and inter-frame space (IFS) [6], [7], [8], [9]. In particular, a selfish user may manipulate the MAC parameters to wait a shorter (backoff) time for transmission than well-behaving users, and thus increases his chance of winning the contention for channel access. Such misbehaving users are shown to be able to capture most available network resources, seriously degrading other well-behaving users' performance [6].

While a variety of solutions have been proposed to address the problem of detecting misbehaving users in 802.11 networks [6], [7], [8], [9], [10], [11], [12], [13], [14], most existing approaches employ behavior-based anomaly detection. Their key idea is to monitor a node's communication behavior and determine whether the behavior follows a legitimate pattern or not. For instance, a monitoring node (e.g., access point) observes the inter-transmission backoff time value of a target node and verifies whether it follows the pattern predicted on the basis of 802.11 protocol [7], [8], [9].

Although these approaches provide a useful insight in the detection of nodes' misbehavior, there are several technical issues that limit their applicability. First, their detection performance hinges on an unrealistic assumption that the transmitter's backoff time information is available to the receiver (i.e., the monitoring node). In 802.11 WLANs, however, it is infeasible to obtain accurate backoff values of other nodes' since the 802.11 protocol does not provide the receiver any information on the transmitter's backoff values [8], [15]. Therefore, the detection scheme needs to use an alternative metric that is practically measurable by, or available to, the monitoring node.

Second, most existing approaches are designed only for homogeneous network conditions in which all nodes have the same criterion for detecting misbehavior [30]. However, in reality, nodes are very likely to experience different packet-error rates due to the spatial and temporal variations in link quality. Each node should, therefore, have a different criterion for the misbehavior detection, depending on its own state, such as per-link transmission-error rate. A good detection scheme must be able to accurately determine individual decision metrics by capturing individual states

- *J. Choi is with the Department of Software Design & Management, Kyungwon University, Soojeong-gu, Seongnam 461-701, Korea. E-mail: jchoi@kyungwon.ac.kr.*
- *A.W. Min is with the System Architecture Lab, Intel Labs, 2111 N.E. 25th Avenue, Hillsboro, OR 97124. E-mail: alexander.w.min@intel.com.*
- *K.G. Shin is with the Real-Time Computing Laboratory (RTCL), Department of Electrical Engineering and Computer Engineering, The University of Michigan, 2260 Hayward St., Ann Arbor, MI 48109-2121. E-mail: kgshin@eecs.umich.edu.*

Manuscript received 10 Feb. 2010; revised 27 Sept. 2010; accepted 4 Nov. 2010; published online 21 Dec. 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2010-02-0061.
Digital Object Identifier no. 10.1109/TMC.2010.262.

(i.e., each link's transmission-error probability) instead of using the average network-wide probability.

Lastly, several statistics-based misbehavior detection algorithms proposed recently [7], [9]. Although they provide better detection accuracy, they require complex calculations for constructing and comparing probability distributions. As a result, they are not suitable for resource-constrained monitoring nodes, such as commercial 802.11 access points (APs).

To overcome the above limitations of existing approaches, we propose an accurate, lightweight, and practical passive-detection mechanism that operates at each AP in 802.11 WLANs. Unlike the existing schemes that require inaccessible hardware-level information (e.g., backoff time), our mechanism exploits the driver-level information that is readily available at the off-the-shelf AP [3]. In particular, the AP acquires all the necessary information from successfully received packets, such as the number of transmissions from each of its clients in between the AP's two consecutive transmissions, and the information included in the MAC headers.

1.1 Contributions

The main contributions of this paper are three-fold as follows:

- Introduction of a new packet-level metric based on the sequence of successfully received packets for the detection of misbehaving nodes in 802.11 WLANs. Unlike the existing backoff-time-based schemes, our scheme requires only the packet-based information, which is readily available at the off-the-shelf AP. We derive a simple, yet accurate closed-form packet-level detection criterion characterizing a legitimate node's behavior, which enables the monitoring node to quickly locate the misbehaving nodes with high accuracy. (see Section 3).
- Development of a lightweight runtime detection algorithm using the sequential hypothesis testing [16]. Our algorithm detects misbehaving nodes on the basis of passive observations at runtime, i.e., successfully received uplink packets from client nodes, without incurring any extra overhead. Our scheme is also much simpler than the statistics-based approaches [7], [9], [12], as it does not rely on the distribution of packet arrivals (see Section 4).
- In-depth analysis and simulation of realistic heterogeneous link scenarios where different links experience different packet-error rates. We develop a simple way of enabling the receiver to accurately estimate the packet-error probability of each individual link, using only a sequence of successfully received packets (see Section 4.2). Our evaluation results show that this approach achieves high detection accuracy under realistic heterogeneous channel conditions and even in the presence of multiple cheating nodes.
- Implementation and evaluation of a prototype of the proposed detection scheme. We demonstrate the practicality of our approach by implementing the proposed detection and passive packet error rate estimation schemes on the *MadWifi* driver [3].

In summary, our detection scheme is simple, accurate, and thus, easy to implement and deploy. We demonstrate that the proposed scheme, despite its simplicity, is highly accurate in detecting selfish behavior.

1.2 Related Work

Even through there is a plethora of work on the detection of misbehaving users in CSMA networks [6], [7], [8], [9], [10], [11], [12], [13], [14], to the best of our knowledge, this is the first online detection method that relies only on easy-to-obtain packet-based information. Most existing detection schemes [6], [8], [14] in 802.11 are designed under the assumption that the backoff-time information of each tagged node is available at the monitoring node. However, it is infeasible to obtain the accurate value of backoff chosen by other stations in 802.11 WLANs. Although there exists alternative methods [7], [9] that calculate the sender's backoff value indirectly from the interarrival time based on the measurement of the channel idle period, they assume that the channel activities of other nodes are perfectly observable and the clocks of nodes are synchronized. However, current network-card drivers do not allow access to the intermediate timing information—such as channel busy and idle periods between two consecutive packet arrivals—required to calculate the accurate value of idle slots [17]. Moreover, the busy channel periods due to the collided or corrupted transmissions are hidden from other nodes and thus cannot be accurately measured.

Recently, an interesting result was reported in [12], where the packet interarrival (system) time—instead of backoff-time—is directly used to identify/detect a misbehaving node. Their detection method is based on the analytical characterization of packet interarrival time distribution at each node. This method overcomes the difficulty in measuring backoff times, but still has several practical issues to be resolved. First, the distribution of packet interarrival time changes dynamically depending on the number of active nodes and their transmission activities. Usually, a variety of packet sizes and transmission rates (up to 54 Mbps in case of 802.11g) are used and the number of active nodes varies with time. However, the method in [12] only considered the RTS/CTS access mechanism in which the collision time is fixed to be the RTS duration[1] and assumes that data transmission times of all nodes are fixed and the number of active nodes is known a priori. This implies that, unlike the analysis results in [12], the packet interarrival time distribution is shown to be continuous, thus exhibiting large differences in the packet interarrival time even among well-behaving nodes. Second, the method requires accurate network status information, such as the collision probability, to calculate the decision criterion (i.e., packet interarrival time distribution) for detecting a misbehaving node. Nevertheless, no operational online method was provided to estimate the network condition and instead, a static network is assumed where both the number of nodes and the expected throughput of a well-behaving node are given a priori. For the above reasons, it is difficult to use the detection mechanism in [12]. By contrast, our proposed online algorithm overcomes these problems by 1) relying on discrete events, such as packet arrivals and departures, decoupled with the packet length and transmission rates, and 2) capturing the dynamics of network condition with a passive estimation technique.

1. In the 802.11 basic access mechanism without any RTS/CTS transmission—the typical usage mode of most 802.11 WLANs—the collision time is unpredictable since it varies with the packet length and transmission rates of nodes.

In order to improve detection accuracy, there have been several recent research efforts [7], [9], [12] utilizing statistics tools. They utilized statistical testing techniques, such as the sequential probability ratio test (SPRT) [7], [12], [18] and Kolmogorov-Smirnov (K-S) test [9]. For example, the method introduced in [9] compares the empirical distribution function obtained from the data samples with a hypothesized legitimate cumulative distribution function (c.d.f.), and identifies misbehaving nodes. However, most of these approaches require to construct the probability distributions from long-term observations, which is too complex to implement in practice. In contrast, our scheme is much simpler than the statistics-based approaches [7], [9], [12], since it uses only a simple closed-form detection criterion without requiring the construction for complex probability distributions.

1.3 Paper Organization

The remainder of the paper is organized as follows: Section 2 describes the system model and overviews our proposed approach. Section 3 introduces a new packet-level legitimate metric, and describes how to collect the statistical information required to verify the behavior of individual nodes. Section 4 details the proposed algorithm and we evaluate its performance via simulation in Section 5, and experimentation in Section 6. We conclude the paper and present our future directions in Section 7.

2 System Model and Proposed Approach

In this section, we first present the system model and the assumptions to be used, and then overview the proposed misbehavior-detection approach.

2.1 System Model

We consider the common IEEE 802.11 infrastructure WLAN consisting of an AP and a set N of client nodes that access the Internet via the AP. The client nodes send their packets to the AP (i.e., uplink transmissions) and the AP forwards the packets to local destinations and/or to remote destinations via the wireline Internet (i.e., downlink transmissions). We assume that APs can be fully trusted since APs usually are maintained by well-trained network administrators. We focus on scenarios where selfish nodes manipulate the channel-access function of the 802.11 protocol, e.g., using smaller CW_{min}, CW_{max}, and IFS (interframe space) than those of well-behaving nodes. In 802.11, it is relatively easy to manipulate the channel-access function, but their detection is not trivial. We are primarily interested in a saturated network condition, because misbehaving nodes can otherwise make insignificant impacts and can thus be ignored. Although we do not consider mitigation of malicious attacks that target to disrupt the network functionality, our solution can be readily applied to the detection of attacks, such as denial-of-service (DoS) attacks.

2.2 Overview of the Proposed Architecture

Fig. 1 depicts a high-level architecture of the proposed system. The proposed detection mechanism is placed at the driver in the AP which functions as a monitoring node. To *quantitatively* characterize each node's behavior, the AP counts the number of incoming packets received from a node in between its two consecutive successful outgoing transmissions, which are readily available at the network-card driver (Section 3). The AP also estimates each node's packet-error probability *passively* based only on the received packets, particularly using the *retry* information [15] in the 802.11 MAC headers (Section 4.2). Based on the estimated error probability, the AP calculates and updates the packet-level metric for the detection of misbehaving nodes (Section 3). Note that the AP processes all of the downlink/uplink traffic delivered from/to its clients in infrastructure-based WLANs, acquiring the data necessary for the detection of misbehaving nodes without incurring any extra overhead. Finally, based on the thus-obtained data, the AP verifies the behavior of each node using a sequential hypothesis test (Section 4.3).

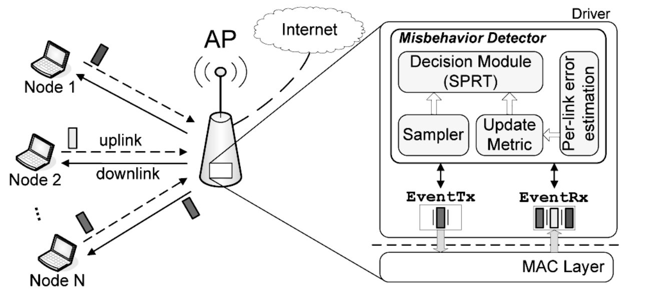


Fig. 1. The system architecture and the proposed detection framework.

3 New Lightweight Detection Metric

In this section, we present a simple and practical metric, namely, the number of intertransmissions, that characterizes the legitimate behavior of 802.11 nodes. We then derive a closed-form expression for the detection metric to *quantitatively* characterize each client node's behavior in an 802.11 WLAN. Finally, we describe how to identify misbehaving nodes using the detection metric.

3.1 Number of Intertransmissions

The key feature of our approach is to exploit the distribution of the number of intertransmissions—the number of packets transmitted by a *target node* between two consecutive transmissions of a *reference legitimate node*—as the main criterion for misbehavior detection. As mentioned earlier, the number of intertransmissions is a packet-level information readily measurable at the AP. Moreover, this simple packet-level metric provides *sufficient* information to the AP in deciding on a node's behavior. Intuitively, in order to achieve higher throughput, selfish nodes must have a larger number of interpacket transmissions than well-behaving nodes within the same time interval, thus making their intertransmission distributions deviate from the distribution of well-behaving nodes.

We have examined the impact of the manipulation of 802.11 MAC parameters on the number of intertransmissions via *ns-2* simulation. As shown in Fig. 2a, the simulation considers three different types of MAC manipulation; 1) CW manipulation (cheating on CW_{min}), 2) IFS manipulation (cheating on DIFS), and 3) binary exponential backoff (BEB) manipulation (cheating on CW_{max}),[2] by setting the MAC parameters (CW_{min}, CW_{max}, DIFS) of a

2. BEB manipulation can be easily realized by setting CW_{max} to CW_{min}, implying that CW is not doubled even upon failure of a transmission attempt due to the small value of CW_{max}.

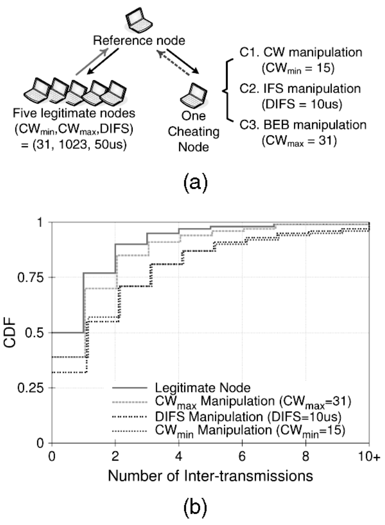


Fig. 2. The number of intertransmissions; (a) simulation scenario, and (b) cumulative distribution function (CDF) of the number of intertransmissions for various types of MAC manipulation.

selfish node to 1) $(15, 1023, 28\ \mu s)$, 2) $(31, 1023, 10\ \mu s)$, and 3) $(31, 31, 28\ \mu s)$, respectively.

The simulation topology in Fig. 2a consists of five legitimate nodes and one selfish node where all these nodes are transmitting packets to a reference node, i.e., the AP. Fig. 2b shows the c.d.f. of the number of intertransmissions. As expected, the selfish node is shown to have higher probabilities for larger interpacket transmissions than the legitimate nodes, indicating that it attempts to access the medium more frequently than the legitimate ones. In what follows, we will detail how the AP detects selfish nodes by using the packet-level information.

3.2 Distribution of Intertransmissions in WLANs

To characterize the behavior of legitimate and selfish nodes, we first derive the distribution of the number of intertransmissions in 802.11 WLANs.

3.2.1 Distribution of Intertransmissions between Two Legitimate Nodes

Let us consider two well-behaving nodes u and l $(u, l \in N)$ following the rule of 802.11 DCF under a saturated condition (i.e., always have packets to transmit). Without loss of generality, we consider node l as the reference node.

Let $K_{u|l}$ denote the number of packets transmitted by node u between two consecutive transmissions of the reference node l. For example, $K_{u|l} = k$ indicates that node u transmits k packets between two consecutive successful transmissions of the reference node l. Our objective is to derive the probability distribution, denoted by $Pr(K_{u|l} = k)$. This intertransmission distribution can be expressed as

$$\begin{aligned} Pr(K_{u|l} = k) &= Pr\left(\sum_{j=1}^{k} T_u(j) \le T_l \text{ and } \sum_{j=1}^{k+1} T_u(j) > T_l\right) \\ &= \sum_{x=0}^{\infty} Pr\left(\sum_{j=1}^{k} T_u(j) \le x \text{ and }\right. \\ &\quad \left.\sum_{j=1}^{k+1} T_u(j) > x \mid T_l = x\right) \cdot Pr(T_l = x), \end{aligned} \tag{1}$$

where T_l denotes the random variable representing the total number of virtual slots (i.e., idle, busy, and failed slots [19]) required for a successful transmission by node l or the interarrival time of node l in terms of virtual slots. $T_u(j)$ denotes the independent and identically distributed (i.i.d.) random variable representing the number of virtual slots for the jth packet at node u.

In (1), the distribution of the number of virtual slots for a successful transmission at node l, i.e., $Pr(T_l = x)$, can be calculated based on the 802.11's BEB mechanism [15]. However, $Pr(T_l = x)$ is the probability mass function of a weighted sum of independent discrete uniform distributions, which belongs to the class of *trapezoidal* distributions. No simple closed-form expressions are known to exist for such distributions [20]. We overcome this difficulty by utilizing an accurate approximation of the intertransmission distribution as we discuss next.

3.2.2 Derivation with Decoupling Approximation

We derive the intertransmission distribution in (1) by using the decoupling approximation introduced in [21]. To make analysis tractable without losing key insights, we make the following three assumptions.

A1. The backoff process of a tagged node l is independent of the other nodes' aggregate transmission attempts.
A2. The packet-transmission attempts by a tagged node l experience a constant and independent error probability p_l.
A3. Node l attempts in each slot with a constant (state-independent) probability equal to the average attempt rate τ_l.

Under these assumptions, we can show that, conditioned on a tagged node's transmission attempt, the number of attempts by other nodes is binomially distributed.[3]

The interarrival time T_l is then geometrically distributed with parameter $\tau_l(1 - p_l)$, i.e., the probability of a successful transmission in a given slot, where τ_l is the average attempt probability. Thus, $Pr(T_l = x)$ can be computed as

$$Pr(T_l = x) = \left\{1 - \tau_l(1 - p_l)\right\}^x \cdot \tau_l(1 - p_l). \tag{2}$$

Here, the average attempt probability τ_l [21] for well-behaving node l with packet-error probability p_l is given by

$$\tau_l = \frac{E[R]}{E[B]} = \frac{1 + p_l + p_l^2 + \cdots + p_l^r}{b_0 + b_1 p_l + b_2 p_l^2 + \cdots + b_r p_l^r}, \tag{3}$$

where $E[R]$ and $E[B]$ denote the average number of attempts required for a packet transmission and the average time spent b_i on backoff before the ith attempt, respectively. Here, b_i is $\frac{2^i CW_{min}}{2}$ for backoff stage i and r denotes the maximum retry counter, typically set to 4. Since the attempt probability τ_l is a function of the error probability p_l as shown in (3), it can be easily calculated for a given value of p_l; the estimation of error probability will be discussed in

3. Note that such simplification has been utilized widely in the analysis of 802.11 WLAN performance under stochastic assumptions [19], [22], [23], [24]. Such an approach is known to yield very good results, capturing important performance attributes, such as throughput and delay in saturated 802.11 networks [19], [22].

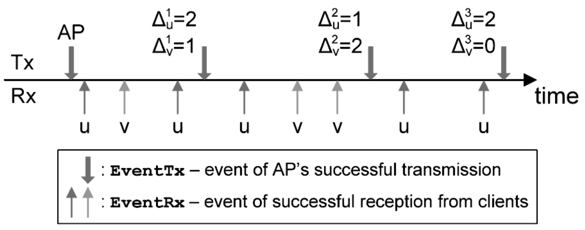

AP
Δ_u^1=2
Δ_v^1=1
Δ_u^2=1
Δ_v^2=2
Δ_u^3=2
Δ_v^3=0
Tx
Rx
time
u
v
u
u
v
v
u
u
: EventTx – event of AP's successful transmission
: EventRx – event of successful reception from clients

Fig. 3. Example measurement and collection of intertransmissions at the AP.

Section 4.2. Then, we can compute $Pr(K_{u|l} = k)$ based on (1) and (2) for the given values of p_l and p_u as

$$Pr(K_{u|l} = k) = \sum_{x=k}^{\infty} \binom{x}{k} \{1 - \tau_u(1 - p_u)\}^{x-k} \times \{\tau_u(1 - p_u)\}^k \cdot Pr(T_l = x). \tag{4}$$

The distribution $Pr(K_{u|l} = k)$ for the number of intertransmissions in (4) characterizes the legitimate behavior of saturating 802.11 node when the node follows the rule of 802.11 protocol. That is, we can identify a misbehaving node by checking whether the observed behavior of a node follows this legitimate distribution or not. In the following sections, we describe how to characterize misbehaving nodes based on the probability distribution.

3.3 Measurement of Client's Behavior

As a first step for detecting misbehaving nodes, the AP monitors and measures the behavior of each individual client node, particularly the sequence of intertransmission number, corresponding to $K_{u|l}$, of the client node $u \in N$. To this end, it is important for the AP to select an appropriate reference node corresponding to node l ($l \in N$) in (4), which follows the 802.11 rule legitimately.

The key idea in selecting such a reference node is that the AP uses its outgoing transmission as the legitimate behavior, i.e., the AP employs itself as the reference node l, i.e., $l = ap$. The rationale behind this approach is as follows: The AP, i.e., the monitoring node, is also an 802.11 node contending for accessing a channel along with its client nodes based on 802.11 DCF. The AP has backlogged packets to transmit when the capacity of the WLAN segment is lower than the wireline bandwidth. Moreover, the AP can, in general, be trusted since it is more powerful and maintained by well-trained network administrators. Therefore, the AP is the most appropriate reference node in a WLAN. We will henceforth regard the AP as the reference node l and focus on its operation, omitting the index ap for notational simplicity, e.g., K_u will denote $K_{u|ap}$.

The AP can obtain the sequence of individual client nodes' intertransmissions by simply counting the number of *incoming (uplink) transmissions* from the client nodes in between its two consecutive successful *outgoing (downlink) transmissions*. Fig. 3 illustrates an example measurement scenario. Here, $\{\Delta_u^i\}_{i=1}^n$ denotes the sequence of intertransmission numbers from node u observed at the AP where the ith element Δ_u^i is the number of intertransmissions between the ith and $(i+1)$th successful downlink transmissions. One key feature of this method is that it is based on the discrete events of packet arrivals/departures and thus is decoupled from continuous communication variables such as packet sizes and transmission rates [12]. As a result, it is simple but accurate in obtaining the samples. Moreover, it does not require any assistance from other protocol layers nor modifications to the 802.11 standard.

3.4 Criterion for Deciding on Misbehavior Using a Simple Closed-Form Detection Metric

We now present a new and practical decision criterion for misbehavior detection by deriving a simple closed-form detection metric.

One can identify a misbehaving node by comparing the empirical distribution of observed sequence $\{\Delta_u^i\}_{i=1}^n$ with the legitimate distribution derived in (4) based on certain well-known goodness-of-fit tests, such as the Kolmogorov-Smirnov test [7], [9]. Although the distribution-based approach is general enough to capture the various types of selfish behavior, it is expensive due to its need for constructing and comparing the distributions over a broad range. Besides, it is nontrivial to compute the legitimate probabilities for higher values of k (e.g., $k > 2$) in (4).

A key distinct feature of our technique is that we rely only on two legitimate probabilities $Pr(K_u = k)$ for $k = 0$ and 1, i.e., $Pr(K_u = 0)$ and $Pr(K_u = 1)$, without the need for the complex calculation of $Pr(K_u = k)$ over the wide range of k. More importantly, this feature enables us to derive a closed-form expression of the detection metric, which is essential for online (runtime) detection.

3.4.1 Characterization of Misbehavior

We first show that the empirical distribution function of the observed sequence, $\{\Delta_u^i\}_{i=1}^n$, of a selfish node deviates from the legitimate distribution in (4). In particular, we show that a selfish node has a higher empirical probability of $Pr(\Delta_u > 1)$, the probability that the number of node u's intertransmissions in $\{\Delta_u^i\}_{i=1}^n$ is larger than 1, than the legitimate probability, $Pr(K_u > 1)$.

We have the following proposition on the intertransmission counts of selfish behavior.

Proposition 1 (Intertransmission count distribution for selfish behavior). *Let $G_u^0(x_0)$ denote the probability that the number K_u of node u's intertransmissions is larger than x_0, i.e., $G_u^0(x_0) = Pr(K_u > x_0)$. Let p_u and p_{ap} denote the given measured error probabilities of node u and AP, respectively. Then, for a given i.i.d. sequence of sample intertransmission counts $\{\Delta_u^i\}_{i=1}^n$, $Pr(\Delta_u > 1) > G_u^0(1)$ if node u is a selfish node.*

Proof. Let τ_{ap} and τ_u^0 denote, respectively, the legitimate attempt probabilities of the AP and node u, which are determined by the 802.11's BEB rule and obtained from (3) for the given p_u and p_{ap} [21]. If node u is well-behaving, its probability distribution of the observed intertransmission counts follows the legitimate distribution derived in (4). Thus, the probability $G_u^0(x_0 = 1)$ can be calculated using $Pr(K_u = 0)$ and $Pr(K_u = 1)$, i.e., $G_u^0(x_0 = 1) = Pr(K_u > 1) = 1 - (Pr(K_u = 0) + Pr(K_u = 1))$, which are obtained from (2) and (3) as follows:

$$Pr(K_u = 0) = \frac{\tau'_{ap}}{1 - \left(1 - \tau_u^{0'}\right)\left(1 - \tau'_{ap}\right)}, \tag{5}$$

$$Pr(K_u = 1) = \frac{\tau_u^{0'} \tau'_{ap}(1-\tau'_{ap})}{\left(1-\left(1-\tau_u^{0'}\right)(1-\tau'_{ap})\right)^2}, \qquad (6)$$

where τ'_{ap} and $\tau_u^{0'}$ denote, respectively, the probabilities of successful transmissions by the AP and node u, and are given as $\tau'_{ap} = \tau_{ap}(1-p_{ap})$ and $\tau_u^{0'} = \tau_u^0(1-p_u)$. Then, we can obtain

$$G_u^0(1) = \left(\frac{\tau_u^{0'}(1-\tau'_{ap})}{1-\left(1-\tau_u^{0'}\right)(1-\tau'_{ap})}\right)^2. \qquad (7)$$

If node u is selfish, then it will have a higher attempt probability, given by $\tau_u^g = \frac{1}{1-g}\tau_u^0\ (\geq \tau_u^0)$, where g is referred to as *Selfish Intensity*, $0 \leq g < 1$, and a larger g implies a higher degree of selfishness. Let $G_u^g(1)$ denote the probability that the selfish node's intertransmission count is larger than 1. Then, the probability distribution of the selfish node's observed behavior will follow this probability, i.e., $Pr(\Delta_u > 1) \sim G_u^g(1)$. We can easily derive $G_u^g(1)$ by substituting τ_u^g for τ_u in (7)

$$G_u^g(1) = \left(\frac{(1-g)^{-1}\tau_u^{0'}(1-\tau'_{ap})}{1-\left(1-(1-g)^{-1}\tau_u^{0'}\right)(1-\tau'_{ap})}\right)^2. \qquad (8)$$

Then, we can show that

$$\sqrt{\frac{G_u^g(1)}{G_u^0(1)}} = \frac{\tau'_u + \tau'_{ap} - \tau'_u\tau'_{ap}}{\tau'_u + (1-g)\tau'_{ap} - \tau'_u\tau'_{ap}} > 1,$$

that is, $Pr(\Delta_u > 1) \sim G_u^g(1) > G_u^0(1)$ for $0 \leq g < 1$. □

Proposition 1 indicates that the AP can obtain sufficient information necessary for decision-making from the observed sequence $\{\Delta_u^i\}_{i=1}^n$ of node u and further verify its behavior by using the legitimate probability $G_u^0(1)$ as a decision metric. In particular, the deviation of its intertransmission distribution from the legitimate probability enables the AP to conclude that the node is selfish.

3.4.2 Decision Criterion for Hypothesis Testing

We can cast the detection problem as a hypothesis testing with two hypotheses, H_0 and H_1, representing the null and alternative hypotheses that the observed node is legitimate and misbehaving, respectively.

Based on Proposition 1, we state the hypothesis testing problem as

$$\begin{cases} H_0 : Pr(\Delta_u > 1) \leq G_u^0(1) & \text{(not misbehaving)}, \\ H_1 : Pr(\Delta_u > 1) > G_u^0(1) & \text{(misbehaving)}. \end{cases} \qquad (9)$$

Here, the probability, $G_u^0(1)$, that the number of node u's intertransmissions K_u is larger than 1, represents the detection metric (i.e., legitimate probability) with which the AP determines whether its client node u is legitimate or misbehaving.

3.4.3 Closed-Form Detection Metric

In the hypothesis test in (9), the detection metric $G_u^0(1)$ is derived as a closed-form function of two transmission error probabilities of the AP and node u, i.e., p_{ap} and p_u, as

$$G_u^0(1) = \left(\frac{\tau'_u(1-\tau'_{ap})}{1-(1-\tau'_u)(1-\tau'_{ap})}\right)^2 = f(p_{ap}, p_u), \qquad (10)$$

where

$$\tau'_l = (1-p_l) \cdot \frac{1 + p_l + p_l^2 + \cdots + p_l^r}{b_0 + b_1 p_l + b_2 p_l^2 + \cdots + b_r p_l^r}, \quad l \in \{ap, u\}.$$

In order to obtain the detection metric for node u in real time, we only need to measure the individual conditional transmission error probability p_u and the reference probability p_{ap} of the AP (see Section 4.2 for details). Note that accuracy in estimating these error probabilities influences greatly the detection performance since it determines the accuracy of the detection metric $G^0(1)$.

4 Passive Online Detection Algorithm

In this section, we present an online algorithm to detect misbehaving nodes in WLANs based on our analysis discussed so far. We first discuss the practical challenges in detecting misbehavior, and then present a practical runtime solution to them.

4.1 Practical Challenges

We identify two main challenges for misbehavior detection based on (9) and (10).

- *Estimation of Individual Error Probability*. As shown in (10), the criterion for detecting misbehavior varies with each node, depending on its own (location-dependent) transmission-error probability. Thus, the AP needs to estimate the *individual* conditional packet error probability (PER) p_u, rather than the average network-collision probability.[4] However, the main difficulty in estimating the uplink PER of a client node is that the AP can observe only the packets successfully transmitted from the node and cannot directly measure the number of total transmission attempts required to calculate the node's PER.
- *Coping with Temporal Variations*. The random nature of the channel access in 802.11 may exhibit temporarily bursty packet transmissions from a well-behaving node, misdiagnosing a well-behaving node as misbehaving. Thus, the detection mechanism must distinguish such temporal variations of well-behaved nodes from deliberate selfish behavior.

For the first challenge, we propose a simple passive online method to estimate individual nodes' error probabilities by exploiting the 802.11's retransmission mechanism. For the second challenge, we employ a sequential analysis technique [16] to accurately detect (i.e., with a small false-alarm probability) misbehavior with a minimal number of observations.

4.2 Online Derivation of Detection Metric

We first present a simple method for the AP to passively estimate the uplink transmission error probability of each individual client node.

4.2.1 Estimation of Clients' Uplink PERs

As mentioned above, the AP cannot directly measure the number of total transmission attempts, n_u^t, nor the number

4. Note that the overhearing-based estimation utilizing the average number of transmission attempts by nodes or the number of idle slots in the network [9], [25] aims to estimate the network-wide average collision probability.

of failures, n_u^f, of node u. The only observable transmissions from clients are successful transmissions. Note that if the AP knows both n_u^t and n_u^f, the probability p_u is computed as $p_u = n_u^f / n_u^t$.

Our key idea in estimating p_u is to leverage the 802.11's retransmission mechanism, i.e., the correlation between the packet-loss probability and the pattern of *Retry* field in the 802.11 MAC header indicated in successively received frames. The Retry field in the 802.11 MAC header[5] consists of a single bit and indicates if a data or management frame is being transmitted for the first time or is a retransmission. For example, a frame with this field set to zero indicates that the frame is successfully transmitted at its first attempt. On the other hand, when this field is set to 1, the frame is the retransmission of an earlier unsuccessful frame. Thus, a larger fraction of packet receptions with the Retry field set to 1 indicates a higher uplink error probability.

Let C_u^r denote the number of frames received from node u with Retry field $r \in \{0,1\}$ at the AP during a measurement period. Since the probability that a frame is successfully delivered at the jth attempt is calculated as $p_u^{j-1}(1-p_u)$, we can obtain the following relations:

- *Successful reception at the first attempt* $(j = 1)$

$$C_u^0 = n_u^t(1-p_u). \tag{11}$$

- *Successful reception via retransmission* $(2 \leq j \leq r+1)$

$$C_u^1 = n_u^t\big(p_u + p_u^2 + \cdots + p_u^r\big)(1-p_u), \tag{12}$$

where r is the retry limit (typically $r = 4$). Dividing (12) by (11), we obtain

$$p_u^r + p_u^{r-1} + \cdots + p_u - \frac{C_u^1}{C_u^0} = 0. \tag{13}$$

As a result, the AP can calculate the conditional probability p_u using (13) based on the measurement of C_u^1/C_u^0 for node u. It can be further simplified to $p_u = C_u^1/C_u^0$ if $C_u^1 \ll C_u^0$.

4.2.2 Online Calculation of Detection Metric

Since the AP can easily calculate its downlink transmission error probability p_{ap} by using its packet transmission history, the AP obtains the *individual detection metric* $G_u^0(1)$ for node u in (10) in real time.

4.3 Sequential Hypothesis Test for Misbehavior Detection

We now propose an online algorithm that detects the selfish behavior via a sequential hypothesis testing, based on the detection criterion in (9).

4.3.1 Likelihood Ratio Test

For a sequential hypothesis testing, we first define the *likelihood ratio test* [26]. Let $p = Pr(\Delta > x_0 = 1)$ be the probability that the intertransmission count of a node is larger than the reference $x_0 = 1$. Let m ($\leq n$) denote the number of observations whose intertransmission number is larger than 1. If node u is well-behaving, its observed sequence Δ_u of intertransmission count will satisfy $Pr(\Delta_u > 1) \leq G_u^0(1)$ as shown in (9). Therefore, if the hypothesis H_0: $p \leq G_u^0(1)$ is rejected by the observed sequence of node u's intertransmission counts, we can conclude that node u is not well-behaving.

For i.i.d. observations, the likelihood ratio test (LRT) statistic is given as [27]

$$\Lambda \triangleq \frac{\sup_{H_0} L\big(p|\Delta_u^1,\ldots,\Delta_u^n\big)}{\sup_{H_1} L\big(p|\Delta_u^1,\ldots,\Delta_u^n\big)} = \frac{\sup_{0\leq p\leq\theta_0} p^m(1-p)^{n-m}}{\sup_{0\leq p\leq 1} p^m(1-p)^{n-m}}, \tag{14}$$

where $\sup_{H_0} L(p|\Delta_u^1,\ldots,\Delta_u^n)$ is the maximum likelihood that the observed sequence $\{\Delta_u^i\}_{i=1}^n$ is in H_0, and θ_0 denotes the legitimate probability (decision criterion) given as $\theta_0 \triangleq G_u^0(1)$ for node u. In (14), the numerator $\sup_{0\leq p\leq\theta_0} p^m(1-p)^{n-m}$ represents the maximum probability of the observed sequence under the null hypothesis H_0 (i.e., not misbehaving). The denominator $\sup_{0\leq p\leq 1} p^m(1-p)^{n-m}$ corresponds to the maximum probability of the observed sequence over all possible cases or in the alternative hypothesis H_1 (i.e., misbehaving). Note that the maximum likelihood estimator (MLE) in the numerator is $\min\{p,\theta_0\}$ while the denominator has p as the MLE. Thus,

$$\Lambda = \begin{cases} 1, & \text{if } p \leq \theta_0, \\ \frac{\theta_0^m(1-\theta_0)^{n-m}}{p^m(1-p)^{n-m}}, & \text{otherwise.} \end{cases} \tag{15}$$

Thus, a small value of Λ indicates that the alternative hypothesis H_1 is more likely with the observed sequence than the null hypothesis H_0. In other words, the likelihood test will reject H_0 if

$$\Lambda < \frac{1}{M}, \tag{16}$$

where $M \in \mathbb{R}$ is the predefined decision threshold. We will study the impact of M on the detection performance in Section 5.

4.3.2 Online Sequential Test Algorithm

Based on the LRT, we can formulate the misbehavior detection problem for node u as a sequential hypothesis test. Let $\widehat{p} = m/n$ denote the MLE of p, i.e., the ratio of the number of intertransmissions larger than 1, to all observed sequences. For $\theta_0 < \widehat{p} < 1$, the likelihood test rejects H_0 and concludes that node u is misbehaving if

$$\frac{\theta_0^m(1-\theta_0)^{n-m}}{\widehat{p}^m(1-\widehat{p})^{n-m}} < \frac{1}{M},$$

which is

$$n < \frac{m\Big(\log\Big(\frac{\widehat{p}}{1-\widehat{p}}\Big) + log\Big(\frac{1-\theta_0}{\theta_0}\Big)\Big) - \log M}{\log(1-\theta_0) - \log(1-\widehat{p})}. \tag{17}$$

For $\widehat{p} = 1$, we have

$$n > -\frac{\log M}{\log \theta_0}. \tag{18}$$

Algorithm 1 describes the procedure for sequential hypothesis testing for N associated client nodes. In the

5. This field is intended to help the receiving MAC eliminate duplicate frames.

algorithm, we adopt $x_0 = 1$ for the reference number, and hence, use $G_u^0(x_0 = 1)$ in (10) as the legitimate probability θ_0, which is given as a closed-form expression. Thus, it can be easily calculated at runtime. For the observed sequence $\{\Delta_u^i\}_{i=1}^n$, m represents the number of sequences j s.t. $\Delta_u^j \geq 2$ $(j \in \{1, 2, \ldots, n\})$.

Algorithm 1. Online Sequential Hypothesis Test

```
procedure $Initialize()$
  1: $x_0 \leftarrow 1$
  2: for all $u$ such that $u \in N$ do
  3:     $n[u], m[u] \leftarrow 0$
  4:     $C_0[u], C_1[u] \leftarrow 0$
  5: end for
procedure $EventReceiveFROM(u)$
  1: // EventRX: Upon receipt of a new packet $\mathcal{P}_u$ from node $u$
  2: // Counting the number of inter-transmissions $K[u]$
  3: $K[u] \leftarrow K[u] + 1$
  4: // Calculation of legitimate behavior reference for node $u$
  5: $C_0[u] \leftarrow C_0[u] + \mathbf{1}$(retry field of $\mathcal{P}_u = 0$)
  6: $C_1[u] \leftarrow C_1[u] + \mathbf{1}$(retry field of $\mathcal{P}_u = 1$)
  7: if $C_0[u] + C_1[u] \geq$ size_update then
  8:     $p_u \leftarrow$ GetPr($C_1[u]/C_0[u]$) // (13)
  9:     $\theta_u \leftarrow G^0(p_{ap}, p_e)$ // (10)
 10: end if
procedure $EventSucTransmit()$
  1: // EventTX: Upon every successful transmission of the AP
  2: for all $u$ such that $u \in N$ do
  3:     $n[u] \leftarrow n[u] + 1$
  4:     $m[u] \leftarrow m[u] + \mathbf{1}(K[u] > x_0)$
  5:     $n \leftarrow n[u]; m \leftarrow m[u];$
  6:     $\widehat{p} \leftarrow m/n$
  7:     if $n < \frac{m(\log \widehat{p}/(1-\widehat{p})) + log((1-\theta_u)/\theta_u)) - \log M}{\log(1-\theta_u) - \log(1-\widehat{p})}$ then
  8:         reject $H_0$ // node u is misbehaving
  9:     else if $\widehat{p} = 1$ and $n > -\frac{\log M}{\log \theta_u}$ then
 10:         reject $H_0$ // node u is misbehaving
 11:     else
 12:         undetermined; do not reject $H_0$
 13:     end if
 14:     $ActivityCheck(u, \widehat{p})$ // Activity check
 15:     $K[u] \leftarrow 0$
 16: end for
procedure $ActivityCheck(u, \widehat{p})$
  1: // ActivityCheck: check whether node u is non-backlogged
  2: if $K[u] = 0$ and $\widehat{p} < \theta_u \cdot \alpha$ then
  3:     // count the consecutive events of $K[u] = 0$,
  4:     // where $\alpha$ is a check parameter ($\alpha = 0.5$)
  5:     $cn[u] \leftarrow cn[u] + 1$
  6: else
  7:     $cn[u] = 0$
  8: end if
  9: if $cn[u] >$ max_consecutive then
 10:     $n[u], m[u] \leftarrow 0$ // node u is non-backlogged; check again
 11: end if
```

Note that the intertransmission count K_u of node u will be always 0 during node u's nonbacklogged period during which the node has no packets to transmit. It may lead to underestimation of MLE $\widehat{p}$ in Algorithm 1, thus misdetecting selfish nodes. This is prevented by including **procedure** $ActivityCheck(u)$ to check and filter out the nonbacklogged period.

TABLE 1
Parameters Used in Performance Evaluation

Parameter	Value
RetryLimit(m)	4
CW_{min}	31
CW_{max}	1023
$CW_i (1 \leq i \leq m)$	min($2^i CW_{min}$, CW_{max})
aSlotTime	9 μs
SIFS	10 μs
DIFS	28 μs
Data Rate	54 Mbps
PHY header (T_{PHY})	192 bits/12 Mbps
ACK	112 bits/12 Mbps
Payload Size	1000 bytes
M (in Eq. (16))	10^4, 10^5

5 Simulation-Based Evaluation

We evaluate the performance of the proposed detection algorithm via simulation with *ns-2* v2.34 [28] in WLAN environments consisting of one AP and multiple mobile client nodes.

5.1 Simulation Setup

The simulation examines three types of MAC manipulation: 1) CW manipulation (cheating on CW$_{min}$), 2) IFS manipulation (cheating on DIFS), and 3) BEB manipulation (cheating on CW$_{max}$). Selfish nodes manipulate MAC parameters to acquire more channel access time while legitimate nodes conform to the MAC parameters specified in the standard, e.g., $CW_{min} = 31$, $CW_{max} = 1{,}023$, and $\text{DIFS} = \text{SIFS} + 2 * aSlotTime$ ($= 28$ μs). We evaluate the proposed detection scheme using various manipulated MAC parameters to study the impact of selfish intensity (i.e., the aggressiveness of selfish behavior) on the detection performance. Particularly, we test three levels of CW manipulation by using $CW_{min} = 7$ (extreme misbehavior), 15 (intermediate), and 24 (moderate), and two levels of IFS manipulation by using $\text{DIFS} = \text{SIFS} + 1 * aSlotTime$ ($= 19$ μs, i.e., one-slot cheating) and $\text{SIFS} + 0 * aSlotTime$ ($= 10$ μs, i.e., two-slot cheating). We also vary the number of contending nodes, i.e., $N = 2, 4,$ and 7, to test the impact of channel contention on the detection performance. In addition, we evaluate our scheme in the presence of multiple selfish nodes (Section 5.2.3).

We consider the IEEE 802.11g PHY/MAC whose system parameters are listed in Table 1. Data traffic is generated by constant bit rate UDP traffic sources under the assumption of a saturated network condition, i.e., there is more traffic than the network can accommodate. Thus, all nodes, including the AP, always have packets to transmit.

To demonstrate the efficiency of our scheme under realistic heterogeneous channel conditions, we simulate the following two scenarios:

- *Homogeneous condition*. All nodes have the same collision probability without channel errors;
- *Heterogeneous condition*. Each node experiences a different channel-error probability.

TABLE 2
Detection Performance: CW Manipulation with $CW_{min} = 7$ for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	1.00	1.00
	Median # of pkts	11	9
	Median Detection Time	0.06	0.06
$N=5$	Detection Rate	0.997	0.99
	Median # of pkts	10	9
	Median Detection Time	0.07	0.06
$N=7$	Detection Rate	0.993	0.993
	Median # of pkts	11	9
	Median Detection Time	0.09	0.07

TABLE 4
Detection Performance: CW Manipulation with $CW_{min} = 24$ for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	1.00	0.95
	Median # of pkts	145	70
	Median Detection Time	0.38	0.18
$N=5$	Detection Rate	0.983	0.95
	Median # of pkts	308	127.5
	Median Detection Time	1.08	0.46
$N=7$	Detection Rate	0.983	0.883
	Median # of pkts	445	119.5
	Median Detection Time	1.79	0.49

Under the heterogeneous link condition, a selfish node may not achieve throughput noticeably higher than legitimate nodes if the selfish node's link is highly prone to errors, thus making its detection difficult. Nevertheless, a detection scheme must be designed to work properly under realistic heterogeneous scenarios.

We also evaluate the impact of the decision parameter M in (16) on the detection performance; we run the simulation 300 times for each set of selfish intensity, number of client nodes, and decision parameter M.

5.2 Detection Performance

5.2.1 Homogeneous Error-Free Channel

First, we evaluate the performance of the proposed misbehavior detection, assuming the wireless medium is error-free, i.e., no transmission error due to fading. Thus, all nodes experience the same packet transmission error due to collision. We consider the cases with a single selfish node and multiple legitimate nodes.

CW Manipulation. Tables 2, 3, and 4 show the detection results for three selfish intensity levels of CW manipulation, $CW_{min} = 7$, 15, and 24, with different numbers of client nodes, $N = 2$, 4, and 7, which all include one selfish node. The table lists the detection rate, the median number of input packets for detection, and median detection time (seconds) in 802.11g PHY. The results indicate that our scheme achieves high accuracy with the correct detection ratio above 96-100 percent ($M = 10^6$) for all simulated scenarios.

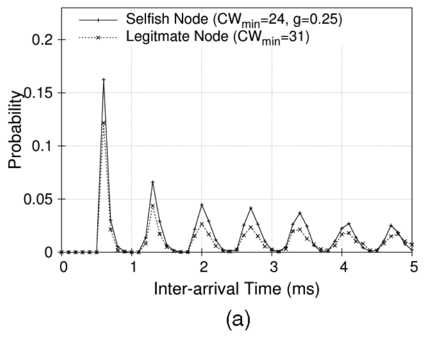
Specifically, when the selfish intensity is high ($CW_{min} = 7, 15$) (Tables 2 and 3), our algorithm is shown to detect the misbehavior very quickly (the median of required observations is less than 30) with high accuracy (more than 99 percent), indicating that more aggressive selfish behavior is likely to be detected more quickly and accurately. This is very important for a detection scheme since such an aggressive behavior can seriously degrade the performance of well-behaving nodes. The results indicate that good detection delay and accuracy for high selfish intensity are maintained even with a large number of clients—the total number of packets required for detection does not change much with a varying number of clients. The results also show the impact of the detection threshold parameter M on the detection performance. A higher value of $M = 10^6$ improves the detection accuracy, but the difference is marginal, i.e., our scheme can detect the selfish behavior of high intensity with very high accuracy even with a small value of M.

On the other hand, the detection of a moderately selfish node (with $CW_{min} = 24$) (Table 4) takes more time, i.e., a larger number of received packets. Specifically, we can observe that the detection delay increases as the number of clients increases. This is because, as the number of nodes increases, the interarrival transmission time for a successful transmission from a client increases due to increased collisions, increasing the time to observe a given number of sequences. Moreover, the impact of such a moderately selfish node on the network performance is not significant, i.e., the selfish node achieves only a small throughput gain over the legitimate nodes. As a result, the moderate selfish node is not immediately detectable at the AP since it takes more samples for the AP to accurately detect such selfish nodes. For example, the simulation result in Fig. 4a shows that the packet interarrival time distribution of a moderately selfish node is similar to that of legitimate node. This implies that the selfish node may be able to achieve only a marginal throughput gain over the legitimate nodes. Hence, it takes more time for the MLE $\widehat{p}$ to satisfy the detection condition in (17) or (18). On the other hand, Fig. 4b shows that the node of high intensity ($CW_{min} = 15$) has a much higher probability in shorter interarrival times under the same condition, which can provide a significant throughput gain to the selfish node.

Unlike the selfish behavior of high intensity with $CW_{min} = 7$ and 15, the result for $CW_{min} = 24$ shows the significant impact of the detection threshold parameter M on the detection accuracy and detection time. There is a trade-off in selecting the detection threshold parameter M; a larger M increases the detection accuracy, but requires a longer detection time.

TABLE 3
Detection Performance: CW Manipulation with $CW_{min} = 15$ for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	1.00	1.00
	Median # of pkts	20	12
	Median Detection Time	0.07	0.05
$N=5$	Detection Rate	1.00	0.993
	Median # of pkts	22	15
	Median Detection Time	0.10	0.06
$N=7$	Detection Rate	0.993	0.98
	Median # of pkts	30	18
	Median Detection Time	0.13	0.09



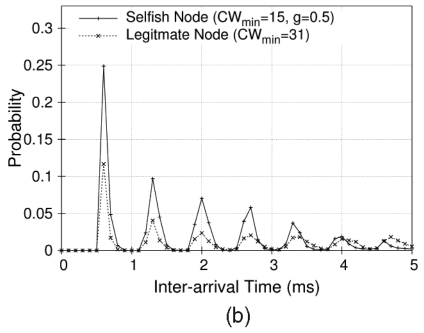


Fig. 4. Distribution of interarrival time under CW_{min} manipulation with different selfish intensities; (a) $CW_{min} = 24$ and (b) $CW_{min} = 15$ for $N = 7$.

DIFS Manipulation. Tables 5 and 6 show the detection results for two selfish intensity levels of IFS Manipulation for $\text{DIFS} = 10$ and $19\ \mu\text{s}$, respectively. When the selfish intensity ($\text{DIFS} = 10\ \mu\text{s}$) is high (Table 5), the proposed scheme is shown to detect the misbehavior very quickly with high accuracy regardless of the number of contending nodes or M. On the other hand, we can observe a decrease in the detection accuracy for a moderate selfish intensity ($\text{DIFS} = 19\ \mu\text{s}$) in Table 6. As in the case of CW manipulation, this can also be explained by the fact that the selfish node with a moderate selfish intensity achieves only a small throughput gain over the legitimate nodes. We can also observe a significant impact of M on the accuracy in detecting low selfishness. It is thus recommended to use a higher value of M, such as $M = 10^6$, to identify a wide range of selfish intensity, although it might take more time to detect.

BEB Manipulation. Table 7 shows the detection performance for BEB manipulation. Note that the BEB cheating affects the selfish node's behavior only when the collision probability is sufficiently high, because unlike legitimate nodes, its gain comes from not increasing its CW value even upon collision. One can see low detection accuracy for a small number of contending nodes, i.e., $N = 2$, because the BEB cheating in this environment does not have a beneficial effect on the throughput gain—the selfish node achieves only a marginal throughput gain (only about 1.13 times) over legitimate nodes. Although the throughput gain of BEB manipulation in the case of a larger number of nodes is shown to be not significant compared to other types of MAC manipulation (i.e., CW and IFS manipulations), we observed that the gain of the BEB cheating node increases as the collision probability increases (about 1.39 and 1.56 for $N = 5$ and 7, respectively). Therefore, the detection accuracy also increases proportionally to the throughput gain.

Overall, our proposed scheme accurately detects various types of MAC manipulation by exploiting only passively observed packet information.

5.2.2 Detection Performance in an Error-Prone Channel

Next, we study the impact of channel error on detection performance under heterogeneous network conditions. Channel errors induce random variations in the number of intertransmissions, K_u, thus affecting the detection performance significantly. We consider a heterogeneous network with one selfish node and two legitimate contending nodes, as shown in Fig. 6. The network is configured so that the two legitimate nodes have fixed uplink packet error rates (PERs) of 0 and 10 percent, respectively. We assume that the AP's downlink transmission is error-free, i.e., $\text{PER}_{ap} = 0$. We performed several experiments while varying PER of the selfish node S for $\text{PER}_S = 0$, 0.15, and 0.25. Note that such link heterogeneity makes the detection challenging. This is because, when a selfish node suffers

TABLE 5
Detection Performance: Interframe Space Manipulation with $\text{DIFS} = 10\ \mu\text{s}$ for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	1.00	1.00
	Median # of pkts	210	149
	Median Detection Time	0.51	0.36
$N=5$	Detection Rate	0.98	0.98
	Median # of pkts	128	100
	Median Detection Time	0.48	0.39
$N=7$	Detection Rate	0.96	0.85
	Median # of pkts	106	98
	Median Detection Time	0.47	0.43

TABLE 6
Detection Performance: Interframe Space Manipulation with $\text{DIFS} = 19\ \mu\text{s}$ for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	1.00	1.00
	Median # of pkts	640	442
	Median Detection Time	1.56	1.02
$N=5$	Detection Rate	0.93	0.62
	Median # of pkts	513	336.5
	Median Detection Time	1.80	1.18
$N=7$	Detection Rate	0.87	0.55
	Median # of pkts	485	286
	Median Detection Time	1.95	1.15

TABLE 7
Detection Performance: BEB Manipulation (Disabled BEB by Setting $\text{CW}_{max} = 31$) for $N = 2$, 5, and 7

		$M=10^6$	$M=10^4$
$N=2$	Detection Rate	0.62	0.75
	Median # of pkts	780	656
	Median Detection Time	1.52	1.59
$N=5$	Detection Rate	0.70	0.76
	Median # of pkts	688	428
	Median Detection Time	1.90	1.55
$N=7$	Detection Rate	0.75	0.73
	Median # of pkts	528	355
	Median Detection Time	1.99	1.66

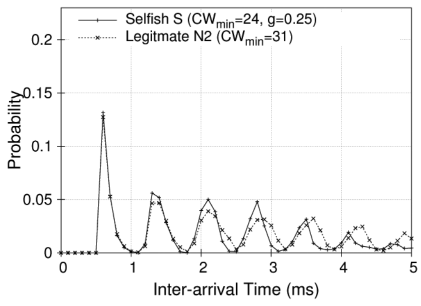


Fig. 5. Distribution of number of packets for detection under selfish intensity of $CW_{min} = 24$ with $\text{PER}_S = 0.25$.

from high PER, the achieved throughput or the number of packets observed at the AP from the selfish node may not be higher or larger than that from legitimate nodes even with a high selfish intensity. For instance, in a scenario with $\text{PER}_S = 0.25$, there is no noticeable difference between the interarrival time distributions of the selfish node S and legitimate node $N2$, as shown in Fig. 5. Thus, it is difficult to identify the selfish behavior based on the interarrival time or achieved throughput [8], [12]. However, a good detection scheme must be robust under channel heterogeneity to be practically useful, since wireless links are usually error-prone due to channel fading, interference, mobility, etc.

In spite of the difficulty under heterogeneous channel conditions, Table 8 shows that our detection scheme achieves high detection accuracy and small delay, being highly efficient under heterogeneous channel conditions.

5.2.3 Detection of Multiple Cheating Nodes

Next, we test the proposed detection scheme in the presence of multiple cheating nodes. With increasing popularity of SDR devices, multiple cheating nodes are likely to coexist in a WLAN. Thus, it is important for a detection scheme to identify such multiple selfish nodes with high accuracy. We consider three scenarios consisting of two, three, and four selfish nodes among five contending nodes ($N_{cheat} = 2$, 3, and 4).

The simulation results in Table 9 show that the detection accuracy of our scheme is very high, as in the single cheating node case. This is because our scheme is based on the independent examination of each individual client node for misbehavior detection, i.e., the AP checks the number of transmissions from each individual client node in between its two consecutive transmissions. An interesting observation in the results in Table 9 is that the detection performance of BEB manipulation ($CW_{max} = 31$) is improved as the number of cheating nodes increases. This can be explained by the fact that the collision probability increases with more cheating nodes due to their selfish behavior, which makes the BEB cheating more effective.

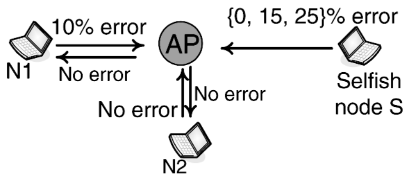


Fig. 6. Simulation topology under heterogeneous error conditions; $\text{PER}_S = \{0.0, 0.15, 0.25\}$.

TABLE 8
Detection Performance under a Heterogeneous Condition for Various Types of MAC Manipulation ($N = 3, M = 10^6$)

		PER=0	PER=0.15	PER=0.25
$\text{CW}_{min}=7$	Detection Rate	0.967	0.993	0.997
	Median pkt #	14	12.5	13
$\text{CW}_{min}=15$	Detection Rate	0.993	1.00	0.99
	Median pkt #	25	31	35
$\text{CW}_{min}=24$	Detection Rate	0.993	1.00	0.993
	Median pkt #	186	304	279
$\text{DIFS}=10\,\mu s$	Detection Rate	0.98	0.99	0.99
	Median pkt #	273	354	465
$\text{DIFS}=19\,\mu s$	Detection Rate	0.91	0.91	0.86
	Median pkt #	667	677	732
$\text{CW}_{max}=31$	Detection Rate	0.71	0.94	0.98
	Median pkt #	925	473	272

Consequently, the increased throughput gain relative to the legitimate nodes makes their detection easier.

6 Implementation and Evaluation

To demonstrate the efficiency of the proposed detection scheme, we have implemented and evaluated a prototype of the scheme by using the MadWifi driver [3] for the AR5212 chipset on Atheros-based WiFi cards on the Linux platform. We used MadWifi because of its stability in implementing an AP mode with the support of `hostapd`. We have performed experiments for one of the cheating schemes based on CW manipulation.

6.1 Accuracy of Passive Per-Link Error Estimation

First, we demonstrate the usability of the proposed passive packet error estimation introduced in Section 4.2 through real experiments. Note that the accurate estimation of a client node's PER is essential to calculate the accurate legitimate metric for the node that dictates the detection performance.

We have implemented this estimation method in the MadWifi driver. When a node receives a frame, the Hardware Abstraction Layer (HAL) calls the frame-hand-

TABLE 9
Detection Performance in the Presence of Multiple Cheating Nodes for Various Types of MAC Manipulation ($N = 5,\ M = 10^6$)

		$N_{\text{cheat}}=2$	$N_{\text{cheat}}=3$	$N_{\text{cheat}}=4$
$\text{CW}_{min}=7$	Miss Detection Rate	0	0	0
	False Alarm Rate	0.04	0.04	0.02
	Detection Accuracy	0.96	0.96	0.98
$\text{CW}_{min}=15$	Miss Detection Rate	0	0	0
	False Alarm Rate	0.05	0.04	0.03
	Detection Accuracy	0.95	0.96	0.97
$\text{CW}_{min}=24$	Miss Detection Rate	0.03	0.09	0.12
	False Alarm Rate	0.11	0.09	0.04
	Detection Accuracy	0.86	0.82	0.86
$\text{DIFS}=10\,\mu s$	Miss Detection Rate	0	0	0
	False Alarm Rate	0.01	0.01	0.01
	Detection Accuracy	0.99	0.99	0.99
$\text{DIFS}=19\,\mu s$	Miss Detection Rate	0	0	0
	False Alarm Rate	0.04	0.02	0.04
	Detection Accuracy	0.96	0.98	0.96
$\text{CW}_{max}=31$	Miss Detection Rate	0.05	0.08	0.07
	False Alarm Rate	0.18	0.10	0.03
	Detection Accuracy	0.77	0.82	0.90

TABLE 10
Backlogged Nodes and Retry Ratio C_1/C_0

N	1	2	3	4
C_1/C_0 from Analysis	0.00	0.062	0.120	0.173
C_1/C_0 from Experiment	0.04	0.09	0.16	0.18

TABLE 11
Legitimate Metric for Online Use in Implementation

p_u	p_{AP}									
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
0	0.23	0.29	0.37	0.46	0.56	0.65	0.74	0.81	0.88	0.94
0.1	0.18	0.24	0.31	0.4	0.5	0.59	0.69	0.78	0.86	0.93
0.2	0.13	0.18	0.24	0.32	0.42	0.52	0.62	0.72	0.82	0.91
0.3	0.09	0.12	0.17	0.24	0.33	0.43	0.54	0.65	0.77	0.88
0.4	0.06	0.08	0.12	0.17	0.24	0.34	0.45	0.57	0.7	0.84
0.5	0.03	0.05	0.07	0.11	0.17	0.25	0.35	0.47	0.62	0.79
0.6	0.02	0.03	0.04	0.07	0.11	0.16	0.25	0.36	0.51	0.72
0.7	0.01	0.01	0.02	0.03	0.06	0.1	0.16	0.25	0.39	0.62
0.8	0	0	0.01	0.01	0.03	0.04	0.08	0.14	0.25	0.47
0.9	0	0	0	0	0.01	0.01	0.02	0.04	0.1	0.25

ling function in the MadWifi driver reporting several transmission statistics, such as the Retry field information. To obtain the ratio in (13), we have inserted the estimation routine and managed this information.

We conducted experiments while varying the number of laptops from 1 to 4, and computed the retry ratio C_1/C_0. Table 10 shows the data obtained from the experimentation. We observed a close match between analysis and experimental results with negligible differences of 0.01-0.03, where the difference is mainly due to the channel errors occurred in real operational WLANs.

6.2 Methodology and Results

For the implementation of our online detection method, we computed and constructed the legitimate metric table offline, corresponding to (10), indexed by the transmission error probabilities of the AP, p_{ap}, and each client node, e.g., p_u for node u. The part of the table is shown in Table 11. At runtime, the detection module on the AP then easily obtains the detection metric, $G^0(1)$, of each individual node by a simple table lookup (without complex numerical computation), using the most up-to-date estimated PERs, $\widehat{p}_{ap}$ and $\widehat{p}_u$, as the index.

We evaluated the experimental results with several values of cheating coefficients of 0.75, 0.5, 0.25, and 0.125, where the cheating coefficient represents the aggressiveness of selfish behavior, defined as the ratio of the legitimate CW value to the manipulated CW value. We built a testbed with one desktop as an AP and two laptops as client nodes. Since currently most commodity network interface cards (NICs) do not allow to tune critical MAC parameters[6] where the functionality of tuning is implemented in the firmware, we implemented the CW manipulation technique by using a bandwidth throttling (traffic shaping) technique in the user space [29] on top of the wireless NIC's driver.

6. Even though MadWifi provides a series of commands (e.g., `iwpriv`) to configure several MAC/PHY parameters, the configuration does not work for some critical MAC parameters including CW_{min}, CW_{max}, and AIFS.

TABLE 12
Experimental Results

Cheating coefficients	0.75	0.5	0.25	0.125
Avg # of Detections	8	14	15	20
Average # of pkts	73.6	66.4	64.9	50

We have differentiated the channel access probabilities of contending nodes in the testbed by allocating different rate limiting parameters. We configured a higher rate at one laptop (which is considered as the selfish node), and same fixed rate at the AP and the other laptop (legitimate node) where we repeated experiments while varying the ratio. In each experiment, we generated 1,000 UDP packets with 500 bytes for each packet and counted the number of detections for pinpointing the misbehaving node. We set the minimum required samples for making decisions to 50 packets, and thus the maximum number of detections for 1,000 UDP packets is bounded by 20. Table 12 shows the experimental results. Our scheme is shown to be able to easily detect the manipulation. When the degree of selfishness increases, the scheme is shown to detect more quickly. Specifically, when the degree of cheating is high (low cheating coefficient), our detection scheme is shown to identify the manipulating node more quickly.

Our implementation on the prototype platform is found to induce negligible overhead in collecting the appropriate measurements and making a decision, thanks to the passive nature of our solution and the table-based implementation.

7 Conclusion

In this paper, we proposed a new, practical detection scheme that relies solely on the sequence of successfully received packets at the AP to detect misbehaving nodes, while most existing schemes require unavailable, hardware-level information, such as backoff time. To further expedite the detection of such misbehaving nodes, we introduced a lightweight online decision algorithm using the sequential hypothesis testing. Our extensive simulation results show that the proposed scheme achieves high accuracy and agility in detecting misbehavior in realistic wireless environments. We also implemented and evaluated the proposed detection and receiver-side estimation schemes using off-the-shelf hardware and the MadWiFi device driver. We plan to apply our passive technique in various scenarios, including identification of hidden nodes in WLANs and discovery of unfairness in 802.11 mesh/relay networks.

Acknowledgments

This work was supported in part by the US National Science Foundation (NSF) under grants CNS-0519498 and CNS-0721529, and Korea World Class University Program under Grant R33-10085.

References

[1] USRP, http://www.ettus.com, 2011.
[2] GNU Software Radio Project, http://www.gnu.org/software/gnuradio, 2011.

[3] Multiband Atheros Driver for WiFi, http://madwifi-project.org, 2011.

[4] M. Neufeld, J. Fifield, C. Doerr, A. Sheth, and D. Grunwald, "SoftMAC—Flexible Wireless Research Platform," *Proc. Fourth Workshop Hot Topics in Networks (HotNets-IV),* Nov. 2005.

[5] J.H. Reed, *Software Radio: A Modern Approach to Radio Engineering.* Prentice Hall, May 2002.

[6] P. Kyasanur and N.H. Vaidya, "Selfish MAC Layer Misbehavior in Wireless Networks," *IEEE Trans. Mobile Computing,* vol. 4, no. 5, pp. 502-516, Sept./Oct. 2005.

[7] S. Radosavac, J.S. Baras, and I. Koutsopoulos, "A Framework for MAC Protocol Misbehavior Detection in Wireless Networks," *Proc. Fourth ACM Workshop Wireless Security (WiSe '05),* Sept. 2005.

[8] M. Raya, I. Aad, J. Hubaux, and A.E. Fawal, "DOMINO: Detecting MAC Layer Greedy Behavior in IEEE 802.11 Hotspots," *IEEE Trans. Mobile Computing,* vol. 5, no. 12, pp. 1691-1705, Dec. 2006.

[9] A.L. Toledo and X. Wang, "Robust Detection of Selfish Misbehavior in Wireless Networks," *IEEE J. Selected Areas in Comm.,* vol. 25, no. 6, pp. 1124-1134, Aug. 2007.

[10] A.B. MacKenzie and S.B. Wicker, "Selfish Users in Aloha: A Game-Theoretic Approach," *Proc. IEEE 54th Vehicular Technology Conf. (VTC-Fall),* Oct. 2001.

[11] M. Čagalj, S. Ganeriwal, I. Aad, and J. Hubaux, "On Selfish Behavior in CSMA/CA Networks," *Proc. IEEE INFOCOM,* Mar. 2005.

[12] Y. Rong, S. Lee, and H. Choi, "Detecting Stations Cheating on Backoff Rules in 802.11 Networks Using Sequential Analysis," *Proc. IEEE INFOCOM,* Apr. 2006.

[13] M.K. Han, B. Overstreet, and L. Qiu, "Greedy Receivers in IEEE 802.11 Hotspots," *Proc. 37th Ann. IEEE/IFIP Int'l Conf. Dependable Systems and Networks (DSN '07),* June 2007.

[14] L. Guang, C.M. Assi, and A. Benslimane, "Enhancing IEEE 802.11 Random Backoff in Selfish Environments," *IEEE Trans. Vehicular Technology,* vol. 57, no. 3, pp. 1806-1822, May 2008.

[15] *IEEE 802.11 WG, IEEE Std 802.11-2007 Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1999,* IEEE, 2007.

[16] H. Poor, *An Introduction to Signal Detection and Estimation.* Springer-Verlag, 1994.

[17] T. Salonidis, M. Garetto, A. Saha, and E. Knightly, "Identifying High Throughput Paths in 802.11 Mesh Networks: A Model-Based Approach," *Proc. IEEE Int'l Conf. Network Protocols (ICNP '07),* Oct. 2007.

[18] A. Wald, "Sequential Tests of Statistical Hypotheses," *The Annals of Math. Statistics,* vol. 16, pp. 117-186, June 1945.

[19] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE J. Selected Areas in Comm.,* vol. 18, no. 3, pp. 535-547, Mar. 2000.

[20] S.M. Sadoogi-Alvandi, A.R. Nematollahi, and R. Habibi, "On the Distribution of the Sum of Independent Uniform Random Variables," *Statistical Papers,* vol. 50, no. 1, pp. 171-175, Jan. 2009.

[21] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New Insights from a Fixed-Point Analysis of Single Cell IEEE 802.11 WLANs," *IEEE/ACM Trans. Network,* vol. 15, no. 3, pp. 588-601, June 2007.

[22] K. Medepalli and F.A. Tobagi, "Towards Performance Modeling of IEEE 802.11 Based Wireless Networks: A Unified Framework and Its Applications," *Proc. IEEE INFOCOM,* Apr. 2006.

[23] M.M. Carvalho and J.J. Garcia-Luna-Aceves, "A Scalable Model for Channel Access Protocols in Multihop Ad Hoc Networks," *Proc. ACM MobiCom,* Sept. 2004.

[24] L. Qiu, Y. Zhang, F. Wang, M.K. Han, and R. Mahajan, "A General Model of Wireless Interference," *Proc. ACM MobiCom,* Sept. 2007.

[25] G. Bianchi and I. Tinnirello, "Kalman Filter Estimation of the Number of Competing Terminals in an IEEE 802.11 Network," *Proc. IEEE INFOCOM,* Mar./Apr. 2003.

[26] M.G. Kendall and A. Stuart, *The Advanced Theory of Statistics.* Griffin, 1973.

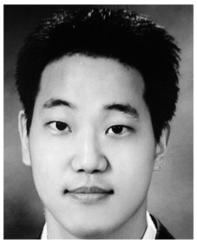
[27] G. Casella and R.L. Berger, *Statistical Inference.* Duxbury Thomson Learning, 2002.

[28] The Network Simulator ns-2 (v2.34), http://www.isi.edu/nsnam/ns, 2011.

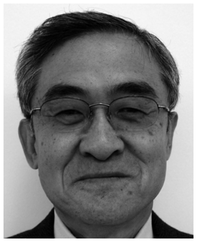
[29] Cisco Systems, "Cisco Tech Notes: Comparing Traffic Policing and Traffic Shaping for Bandwidth Limiting," Document ID: 19645 Graphs Illustrate Differences in Typical Output, 2005.

[30] A. Venkataraman, C.L. Corbett, and R.A. Beyah, "A Wired-Side Approach to MAC Misbehavior Detection," *Proc. IEEE Int'l Conf. Comm. (ICC '10),* May 2010.

Jaehyuk Choi received the PhD degree in electrical engineering and computer science from Seoul National University, Korea, in 2008. He is currently an assistant professor in the Department of Software Design & Management at Kyungwon University, Seongnam, Korea. From 2009 to 2011, he was a postdoctoral research fellow in the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor. He was a postdoctoral fellow in Brain Korea 21 at Seoul National University in 2008. His current research interests are in the areas of wireless/mobile networks with emphasis on wireless LAN/MAN/PAN, network management, next-generation mobile networks, cognitive radios, data link layer protocols, and cross-layer approaches. He is a member of the IEEE.



Alexander W. Min received the BS degree in electrical engineering from Seoul National University, Korea, in 2005 and the PhD degree in electrical engineering and computer science from the University of Michigan, Ann Arbor, in 2011. He is currently a research scientist in the System Architecture Lab at Intel Labs. In 2010, he was a research intern at Deutsche Telekom Inc., R&D Labs, Los Altos, California. His research interests are in the areas of cognitive radio and dynamic spectrum access networks, wireless security, energy-efficient mobile platforms, and mobile sensing. He has served as a reviewer for leading networking journals and conferences and on the technical program committees for IEEE PIMRC and IEEE ICC. He is a member of the ACM and the IEEE Communications Society.



Kang G. Shin received the BS degree in electronics engineering from Seoul National University, Seoul, Korea, in 1970, and the MS and PhD degrees in electrical engineering from Cornell University, Ithaca, in 1976 and 1978, respectively. He is the Kevin & Nancy O'Connor professor of computer science and founding director of the Real-Time Computing Laboratory in the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor. From 1978 to 1982, he was on the faculty of Rensselaer Polytechnic Institute, Troy, New York. He also chaired the Computer Science and Engineering Division in the Electrical Engineering and Comptuer Science Department at the University of Michigan from 1991-1993. He has held visiting positions at the US Airforce Flight Dynamics Laboratory; AT&T Bell Laboratories; the Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley; the International Computer Science Institute, Berkeley, California; the IBM T.J. Watson Research Center; the Software Engineering Institute at Carnegie Mellon University; HP Research Laboratories; the Hong Kong University of Science and Technology; Ewha Womans University, Korea; and the Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. His current research focuses on computing systems and networks as well as on embedded real time and cyber-physical systems, all with emphasis on timeliness, security, and dependability. He has supervised the completion of 69 PhDs and (co)authored more than 770 technical articles. He coauthored (jointly with C.M. Krishna) the textbook *Real-Time Systems* (McGraw Hill, 1997). He has received numerous best paper awards and several institutional awards. He is a member of the Korean Academy of Engineering. He has served as a (co)chair for many conferences, including MobiCom 2009, SECON 2008, MobiSys 2005, RTAS 2000, RTSS 1986, and RTSS 1987 and has served on numerous technical program committees. He also chaired the IEEE Technical Committee on Real-Time Systems from 1991-1993 and was a distinguished visitor of the IEEE Computer Society, an editor of the *IEEE Transactions on Parallel and Distributed Computing*, and an area editor of the *International Journal of Time-Critical Computing Systems*, *Computer Networks*, and *ACM Transactions on Embedded Systems*. He is a fellow of the IEEE and the ACM.