A Fair Transmission Opportunity by Detecting and Punishing the Malicious Wireless Stations in IEEE 802.11e EDCA Network

Yong woon Ahn, Jinsuk Baek, *Member, IEEE,* Albert Mo Kim Cheng, *Fellow, IEEE,* Paul S. Fisher, and Minho Jo, *Member, IEEE*

***Abstract*—We propose a malicious detection algorithm that permits identification of misbehaving wireless stations, and then dispenses punishment by denying an ACK packet which allows transmission by the malicious stations. The proposed algorithm is designed for an IEEE 802.11e network and is based upon detecting a change in quality of service (QoS) where a station is moved up in terms of permission to a level not justified. Non-detection causes honest stations to increase their retransmit attempts, exceeding the specified deadline in order to sustain continuous receipt of their data. Even with the coexistence of such malicious stations, our algorithm guarantees the lower bound of performance degradation to the honest stations. Also, the punished period imposed upon the malicious stations compensates the experienced degradation in QoS for the honest stations in the next transmission period. As a result, we provide fair resource sharing among the stations operating from the same access point. Our simulation within the ns-2 framework of the IEEE 802.11e enhanced distributed channel access network shows how our algorithm actually detects and adjusts the punishment phase for stations that may be misbehaving as well as stations which are misbehaving.**

***Index Terms*—Cheater detection, cheater punishment, IEEE 802.11e, malicious station, transmission opportunity.**

I. Introduction

CONTEMPORARY wireless devices should be equipped with functionality for supporting quality of service (QoS) applications such as voice or video streaming services. Unfortunately, most widely used medium access control (MAC) protocols, IEEE 802.11a/b/g, could not satisfy these requirements because they do not provide assurance for fluent QoS. For example, the distributed coordination function (DCF) in the legacy IEEE 802.11 series could not guarantee performance for multimedia applications because it only works as a contention-based channel coordination function. As an alternative, the point coordination function has been proposed to provide a contention-free period. During this period, the access point (AP) grants a contention-free channel access to each wireless station with the polling mechanism based upon a round-robin scheme. However, we first note that only one polled station can transmit data in each polling interval. In addition, from the AP's perspective, there is no efficient way to determine the volume of data that will be transmitted from the polled station. This causes unpredictable beacon delay, and it does not allow other stations to fairly access the shared medium to transmit their data. More importantly, it does not include any mechanism to prioritize transmissions among the different data flows.

In order to overcome these limitations, IEEE 802.11e has been proposed by the Task Group e (TGe). Unlike the legacy IEEE 802.11 technologies, IEEE 802.11e considers prioritized traffic using both contention and contention-free periods for the wireless stations in the super frame. To support two separate periods, TGe has implemented a hybrid coordination function (HCF) consisting of the HCF controlled channel access (HCCA) function and the enhanced distributed channel access (EDCA) function. For both channel access functions, a new concept, the transmission opportunity (TXOP) has been introduced. That is, during TXOP period, a wireless station can burst its QoS data without any interruption by other wireless stations. For the contention-free period, HCCA is used with the hybrid coordinator (HC) installed at the AP. The HCCA defines a traffic specification (TSPEC) frame describing the QoS requirements for each station including maximum and minimum packet size, maximum and minimum data rate, maximum jitter, and the maximum and minimum packet count. Using the TSPEC frame, each QoS enhanced wireless station (QSTA) negotiates with the QoS enhanced access point (QAP) for acquiring enough TXOP duration for transmission. Fig. 1 depicts the structure of the 802.11e super frame which consists of the contention-free period operated by HCCA, and the contention period operated by both HCCA and EDCA. Every super frame starts with the beacon frame which is periodically broadcast by QAP. The beacon frame includes network parameters which can be used for managing contention among the wireless stations. The polled TXOP with

Manuscript received October 28, 2010; revised March 4, 2011; accepted May 12, 2011. Date of publication September 23, 2011; date of current version November 23, 2011. This work was supported in part by the National Science Foundation, under Award 0720856, and by the National Research Foundation of Korea, under Grant 2011-0009454 funded by the Korean Government.

Y. W. Ahn and A. M. K. Cheng are with the Department of Computer Science, University of Houston, Houston, TX 77204 USA (e-mail: yahn@cs.uh.edu; cheng@cs.uh.edu).

J. Baek and P. S. Fisher are with the Department of Computer Science, Winston-Salem State University, Winston-Salem, NC 27110 USA (e-mail: baekj@wssu.edu; fisherp@wssu.edu).

M. Jo is with the College of Information and Communications, Korea University, Seoul 136-701, Korea (e-mail: minhojo@korea.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSYST.2011.2165598

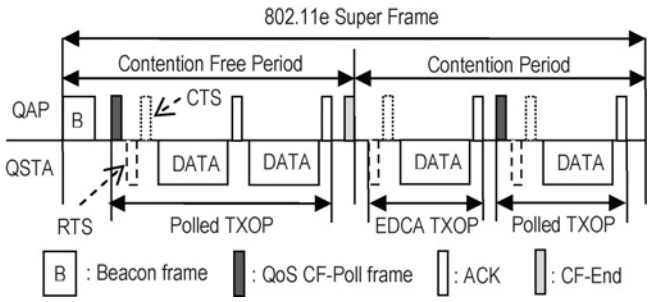


Fig. 1. IEEE 802.11e super frame service.

TABLE I
USER PRIORITY TO AC MAPPING

Priority	User Priority	AC	Destination (Informative)
Lowest	1 or 2	AC_BK	Background
	1 or 3	AC_BE	Best effort
↓	4 or 5	AC_VI	Video
Highest	6 or 7	AC_VO	Voice

TABLE II
DEFAULT EDCA PARAMETER SET

AC	AIFSN[AC]	TXOPLimit (ms)	CWmin[AC]	CWmax[AC]
AC_BK	7	0	CWmin	CWmax
AC_BE	3	0	CWmin	CWmax
AC_VI	2	6.016	CWmin/2	CWmin
AC_VO	2	3.008	CWmin/4	CWmin/2

HCCA can be acquired through the negotiation phase between QSTA and QAP using a TSPEC frame. Once the HC at the QAP determines the TXOP-granted QSTA, it schedules the polled TXOP duration used by the QAP, and informs the QSTA by sending a QoS CF-Poll frame. When the contention-free period ends, QAP broadcasts the CF-End frame to all QSTAs to initiate the contention period with EDCA.

On the other hand, the contention period is not managed by any particular mediator in QAP. To transmit data during the contention period with EDCA, each QSTA competes with other QSTAs based upon the broadcasted EDCA parameters. With the EDCA, the prioritized flows are considered. That is, it confers a higher probability of transmission of data having a shorter inter-frame space. EDCA categorizes QoS applications with eight user priorities. Usually voice applications have the highest priority, and ordinary best effort applications have the lowest priority. These eight user priorities are grouped into four access categories (ACs). Table I shows the mapping relationships between user priorities and ACs. Each QSTA maintains four outgoing packet queues for the four ACs, and each packet would virtually contend with the other packets in other queues having different EDCA parameter values. The EDCA parameters include: 1) the arbitrary inter-frame space number (AIFSN[AC]) which is used to calculate AIFS; 2) the minimum value of the contention window (CWmin[AC]); 3) the maximum value of the contention window (CWmax[AC]); and 4) the maximum duration of TXOP (TXOPLimit[AC]). Table II shows the default EDCA parameter values. Note that the smaller values mean there is a greater opportunity to access the channel.

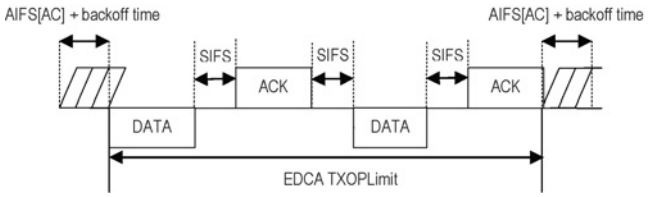


Fig. 2. EDCA TXOPLimit with DATA and ACK packet.

The structure of an EDCA TXOPLimit is shown in Fig. 2. Each QSTA can burst its QoS data during the assigned TXOPLimit period and the value of the TXOPLimit can be determined as either a constant or as a dynamic value [1].

We claim all QSTAs must obey the broadcasted EDCA parameter values including TXOPLimit for fair contention among the QSTAs. Otherwise, if there is one or more malicious QSTAs which intentionally disrupt fair contention and behave selfishly taking more opportunity to access the medium, other honest QSTAs eventually experience performance degradation over time. Unfortunately, all MAC layer protocols including IEEE 802.11e have been developed assuming all wireless stations will honestly obey the suggested network parameter values. This optimistic assumption is more serious for IEEE 802.11e network because it is designed to implement differentiated traffic. As a result, the malicious QSTAs consume more channel throughput than assigned; and obstruct other QSTAs attempting to transmit their timing-sensitive QoS data. In order to cope with this issue, some important parameters should be measured and examined.

In this paper, we first consider possible misbehaviors modifying the parameter values and propose a simple but efficient mechanism to detect the abnormal QSTAs based upon these parameter measurements. Once detected, the QSTAs are suspended in the next transmission period. The amount of suspended time depends on the level of violations. More importantly, our well-defined threshold limits the level of violation, which guarantees the minimum consistent performance to honest QSTAs. Also, this performance degradation at the current transmission period will be sufficiently compensated at the next transmission period.

The remainder of this paper is organized as follows. Section II discusses the related work and the problems with existing approaches. Section III discusses possible misbehaviors of QSTAs, which can be observed in IEEE 802.11e networks. Section IV proposes our scheme as an effort to detect the malicious QSTAs. Section V proposes our punishment algorithm to provide an eventual fair resource sharing among the QSTAs. In Section VI, we show the performance of the proposed algorithm, followed by the conclusions given in Section VII.

II. RELATED WORKS

Recently, many solutions have been proposed to efficiently detect network attacks in a global network environment. One of the examples is in [2]. As our concern is on the wireless local area networks (WLANs), our investigation for relevant work is focused on the schemes to detect the malicious stations in WLANs.

In [3] and [4], the authors investigated a case of a forged backoff value, and proposed a new scheme requiring few

modifications to the DCF used in the IEEE 802.11a/b/g network. The receiver randomly selects the backoff value based on the lower bound which is assigned by the sender. If the sender's backoff time is smaller than the assigned value, the receiver considers that the sender is malicious because a smaller backoff time would eventually provide more opportunity to access the shared channel.

Another approach [5] is extended from Bianchi's model [6]. In order to model the misbehaving wireless stations, it specified some malicious cases where the cheater could fix its contention window. A game-theoretic approach was used to investigate the selfish behaviors with the Nash equilibrium. However, they assumed the network is always in the saturated condition which would be infeasible in the practical situation.

A predictable random backoff (PRB) is proposed in [7]. The key idea of this approach is to adjust the lower bound of the contention window. A 3-D Markov chain is used to compute the system throughput. Although PRB is able to ensure the detection/punishment of malicious/selfish wireless stations, they only consider problematic cases found in mobile, ad hoc networks.

In [8], DOMINO, software to be installed at the access point, is developed. This includes multiple modules to detect various kinds of misbehaviors by wireless stations. However, they could not show cases relevant to EDCA in IEEE 802.11e networks. Although there are many scheduling algorithms [9], [10] for WLANs, to the best of our knowledge, there is no literature focusing on the malicious detection issue for the IEEE 802.11e network.

We recently proposed a detection algorithm [11] for determining the malicious QSTAs but our main focus was to validate if our algorithm operated properly with a simple and straightforward punishment function. Therefore, any judicial punishment policy was not conducted. Accordingly, once the malicious QSTAs were suspended, there was no efficient way to allow the malicious stations with some imposed penalty to go back to their normal operation. In this paper, we turn our attention to design the punishment algorithm providing a fair transmission opportunity among the QSTAs.

III. Possible Misbehaviors

A. Make A Shorter AIFS/DIFS/Random Backoff Time

In IEEE 802.11 networks, the malicious/selfish wireless station/QSTA may forge AIFS/DIFS values to minimize the waiting time, or modify AIFS/DIFS+backoff time to transmit its next data packets with a shorter wait interval. As a result, the station/QSTA can increase the probability of channel access by intentionally minimizing the AIFS/DIFS/backoff time. To cope with this problem, we use the approach proposed by [8] with modifications to add the concept of the AC in EDCA.

B. Make A Longer TXOPLimit

One of the most important concepts in the IEEE 802.11e network is TXOP because all QoS data transmissions should be completed within the assigned TXOPLimit to maintain the desirable QoS level of their voice/video streaming applications.

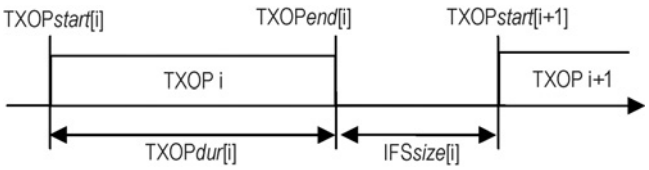


Fig. 3. Time chart with variables used for the proposed algorithm.

As shown in Fig. 2, one TXOP cycle consists of DATA and ACK packet pairs with SIFS time. Once a QSTA acquires TXOP duration, other QSTAs cannot intervene during this duration. Therefore, if a malicious QSTA autonomously increases a value of the TXOPLimit, other honest QSTAs must increase their backoff window, potentially missing their deadline to transmit data. We focus on the cases of forging the TXOPLimit by malicious/selfish QSTAs. There are two different methods to determine TXOPLimit value, namely a static and dynamic method. To use the static TXOPLimit, the QAP simply maintains and adjusts the value of TXOPLimit as a constant value, and broadcasts such adjustments to all connected QSTAs. In contrast, the dynamic TXOPLimit can be calculated with the QoS requirements of each QSTA such as lower bounds of throughput, jitter, or delay. The calculation is proposed in [1]

$$T_i^{\text{required}} = \text{DATA}dur_i n_i + (2n_i - 1)\text{SIFS} + n_i \text{ACK}dur_i \quad (1)$$

where T_i^{required} is the traffic throughput requirement of QSTA i, DATAdur_i is the duration of packet transmission time of the QSTA i based upon the MAC service data unit, n_i is the number of packets of QSTA i, and ACKdur_i is the duration to transmit an ACK frame to QSTA i. As SBA_i is defined as the surplus bandwidth allowance for the expected error performance, the TXOPLimit_i for QSTA i is given by

$$\text{TXOPLimit}_i = T_i^{\text{required}} \text{SBA}_i. \quad (2)$$

The example to determine the default SBA value described in TSPEC is available in [13].

IV. Potential Cheater Detection Mechanism

We first design and implement the malicious behavior detection algorithm. Our algorithm uses recorded values of the slot time for each QSTA. The QAP records statistics for several beacon indexes. In every beacon index, the duration of TXOP (TXOP*dur*) and the inter-frame space size (IFS*size*) are recorded.

In order to calculate TXOP*dur*, the starting time of TXOP and the ending time of TXOP must be measured as depicted in Fig. 3. The QAP checks the destination address of the previously sent ACK packet whenever it receives a DATA packet. If the current DATA packet's source address and the previous ACK's destination address are identical, the QAP recognizes that the TXOP has been initiated. In order to simplify our discussion, we consider that one cycle of transmission includes DATA, SIFS, ACK, and SIFS. That is,

we do not consider the RTS/CTS mechanism [14], which is not mandatory for TXOP usage. A TXOP may consist of the multiple cycles. Even though TSPEC contains the size of the DATA packet's payload defined at the negotiation phase, there could be unexpected link delay by erroneous situations. This consideration is very useful because we can adapt the proposed algorithm to the various bit rate data transmissions as well as the constant bit rate requirements. In every transmission cycle j within beacon i, the QAP calculates the actual duration of the DATA packet, DATA$dur[i][j]$, with

$$\begin{aligned} \text{DATA}dur[i][j] &= \text{ACK}start[i][j] - \text{DATA}arr[i][j] \\ &\quad - \text{SIFS} \end{aligned} \tag{3}$$

where i is the current beacon index while j is the current cycle index. ACK$start$ is the slot time that QAP sends the ACK packet to the destination. DATAarr is the slot time in which the QAP receives the first bit of the DATA packet. DATAdur can include the link delay which is dependent on the DATA payload size described in the TSPEC. Of course, the QAP can calculate duration of one cycle, CYCLEdur, with

$$\begin{aligned} \text{CYCLE}dur[i][j] &= \text{DATA}dur[i][j] + SIFS + \text{ACK}dur \\ &\quad + SIFS \end{aligned} \tag{4}$$

where ACKdur is a constant value determined by the QAP. If the source address of the most recently received DATA packet is different from the address of previously received packet, the QAP considers that TXOP of the previous one has finished, and starts the detection algorithm. The TXOP$dur[i]$ at current beacon i can be now calculated by

$$\text{TXOP}dur[i] = \sum_{j=0}^{c-1} \text{CYCLE}dur[i][j] \tag{5}$$

where c is the total number of cycles, and j is the cycle index. Note that the calculated TXOPdur should not exceed the assigned TXOPLimit. Otherwise, the source QSTA can be considered as a candidate for a malicious/selfish node. This simple consideration is not always true because we need to consider a case of consecutive TXOPs. For example, if two consecutive cycles have the same source address, the algorithm recognizes that the measured TXOPdur is much larger than the assigned threshold TXOPLimit, even if those behaviors are legal and normal. Thus, in order to separate two or more consecutive TXOPs, our algorithm has a module to calculate the inter frame space size, IFS$size$, using

$$\text{IFS}size = \text{currentTime} - \text{ACK}start[i-1] - \text{ACK}dur \tag{6}$$

where currentTime is a current slot time index, ACK$start$ is a slot time index that QAP sends an ACK packet to the corresponding QSTA, and i is the beacon index. The resultant value IFS$size$ can be additionally used to determine if the previous TXOP ends or not with

$$\text{IFS}size < \min\{\text{AIFS}[s][i] \mid 1 \le i \le B\} \tag{7}$$

where s is the identification of the given QSTA. We consider if the calculated IFS$size$ is smaller than any defined AIFS value at the current beacon index B, it is considered as SIFS. In other words, the TXOPmax is still increasing. After calculating TXOPmax the QAP determines if the previous source QSTA is the potential cheater based on

$$\begin{aligned} \text{TXOP}max[s] &\ge \max\{\text{TXOPLimit}[s][i] \mid 1 \le i \le B\} \\ &\quad + \sum Delay \end{aligned} \tag{8}$$

where the second term is the cumulative delay including queuing delay, collision, and on-air delay. Note that each QSTA's s deserves to have at least a TXOPLimit$[s][i]$ transmission opportunity with the corresponding AC. We assume that the cumulative delay can be measured and recorded by the QAP. Once the QSTA satisfies (8), it is considered a potential cheater.

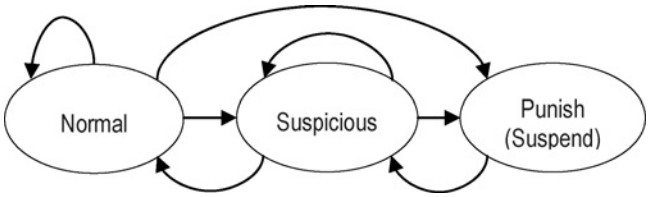


Fig. 4. State transitions to punish the cheater.

V. Imposing Penalty to Actual Cheaters

The next issue is how QAP determines if the potential QSTAs are the actual cheaters and if so, how to punish the QSTAs without significant performance degradation of other honest QSTAs. For determination, we proposed a penalty-based approach. With our algorithm, we define a flag variable, which is set to true when (8) is satisfied. We also define three states of the potential cheater. Fig. 4 shows the three states, namely, normal, suspicious, and punish.

Let us suppose there are N QSTAs under the QAP. In a given beacon index i, the condition defined in (8) is observed. Whenever the condition is satisfied for a QSTA s within a same beacon index i, the accumulated difference AD$[s][i]$ between the two terms is calculated as follows:

$$\begin{aligned} \text{AD}[s][i] &= \text{TXOP}dur[s][i] - \text{TXOP}max[s] \\ &\quad \text{for} \quad 1 \le i \le N. \end{aligned} \tag{9}$$

Here, we define our predefined threshold T$[i]$ in terms of level of violation. Depending on the size of AD$[s][i]$ against the threshold T$[i]$, the transition toward the punish state is performed as follows.

1) *Normal State:*
   If (AD$[s][i] \le 0$) Stay at normal state
   Else If (AD$[s][i] \ge$ T$[i]$) Move forward to punish state
   Else Move forward to suspicious state.
2) *Suspicious State:*
   If (AD$[s][i] \le 0$) Move backward to normal state
   Else If (AD$[s][i] \ge$ T$[i]$) Move forward to punish state
   Else (AD$[s][i] <$ T$[i]$) Stay at suspicious state
   Calculate TXOPLimit[AC][i+1] time for the next beacon index i+1.

3) *Punish State:*
Suspended for specific number of time intervals, PUNISH*dur*, and then move backwards to the suspicious state.

With our algorithm, the threshold value T[i] is the most important factor to decide the actual cheaters and impose the associated penalty. We recognize the large T[i] results in significant performance degradation for honest QSTAs, which is not appropriate for time-sensitive multimedia data transmission. Conversely, the small T[i] minimizes such temporal degradation. However, it is generally difficult to reflect the delay nature of dynamic network.

We need to mention that our goal is providing a lower bound of performance degradation for honest QSTAs even with the malicious QSTAs. That is, our purpose is to allow each honest QSTA to use at least some portion of its full transmission opportunity, although we will eventually guarantee full use of its allocated opportunity.

The temporal performance degradation for honest QSTAs depends upon the number of involved malicious QSTAs and their associated measure of violations against the allocated transmission opportunity. Once the saturated transmission among the N QSTAs is assumed, considering the delay factor as constant, each QSTA s deserves to have TXOPLimit[s][i] transmission opportunity at beacon index i. However, if M malicious QSTAs are involved, each honest QSTA has a reduced transmission opportunity as follows:

$$\text{TXOP}dur[s][i] = \text{TXOPLimit}[s][i] - \frac{\sum_{j=1}^{M} \text{AD}[j][i]}{N-M}. \quad (10)$$

In order to guarantee the lower bound L in percentage units of transmission opportunity, the amount of violation should be limited as follows:

$$L = \frac{\text{TXOP}dur[s][i]}{\text{TXOPLimit}[s][i]} \quad (11)$$

resulting in

$$\sum_{j=1}^{M} \text{AD}[j][i] = \text{TXOPLimit}[s][i](1-L)(N-M). \quad (12)$$

Therefore, the threshold T[i] for each QSTA should be set to

$$\text{T}[i] = \frac{\text{TXOPLimit}[s][i](1-L)(N-M)}{M}. \quad (13)$$

In order to reflect the stateless property at the initial phase, our system sets an appropriate initial value T[0] to the mean value of the medium delay. Afterward, the value T[i] for the next beacon index i has been adjusted when our system detects anomalies, disturbing fair contention among the QSTAs. Note that the number of malicious QSTAs M is available by counting the number of QSTAs in suspicious states. The QSTAs in the punish state should be suspended for "AD[s][i] − T[i]" time slot for current beacon index i. Also, the transmission opportunity TXOPLimit[s][i+1] for the next beacon index i+1 should be set to "TXOPLimit[s][i] − T[i]." On the other hand, for the QSTAs in a suspicious state,

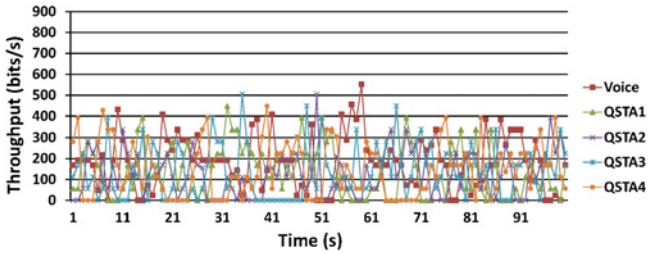


Fig. 5. Throughput (kbits/s) without a cheater.

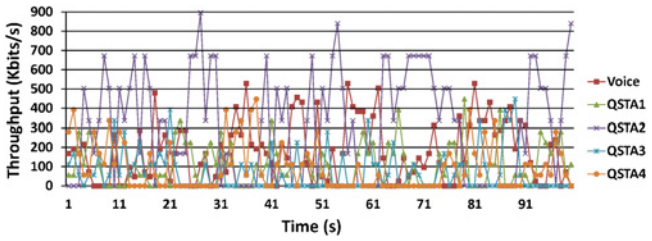


Fig. 6. Throughput (kbits/s) with a cheater, QSTA2.

the transmission opportunity TXOPLimit[s][i+1] for the next beacon index i+1 should be set to "TXOPLimit[s][i] − (T[i]− AD[s][i])." Conversely, for the QSTAs in normal state, when there are M malicious QSTAs with P suspended QSTAs in the current beacon index i, the same should be set to

$$\text{TXOPLimit}[s][i] + \frac{\sum_{j=1}^{P} (\text{AD}[j][i] - \text{T}[i])}{N-P}. \quad (14)$$

In order to fairly reward TXOPLimit values to honest QSTAs for the next beacon index, another important factor for consideration is the length of time the malicious QSTAs should be prevented from transmitting data. That is, how long of a time will they have to stay in the punish state. Therefore, we define the punish duration in terms of the number of beacon index intervals as follows:

$$\begin{aligned} \text{PUNISH}dur[s] &= \frac{\text{AD}[j][i] - \text{T}[i]}{\frac{\sum_{k=1}^{N} \text{AD}[k][i]}{N}} \\ &= \frac{N(\text{AD}[j][i] - \text{T}[i])}{\sum_{k=1}^{N} \text{AD}[k][i]}. \end{aligned} \quad (15)$$

In summary, for the honest QSTAs, the proposed threshold value will limit the level of performance degradation in current beacon i. Moreover, the performance degradation will be compensated for in next beacon index i+1. We need to mention that our goal is to show how our algorithm operates in a multi-stations environment. In actual implementation, the number of states can be extended by defining multiple suspicious states if the network administrator wants to allow more misbehavior states. Also, the lower bound parameter L can be adaptively set depending on the needs of target applications.

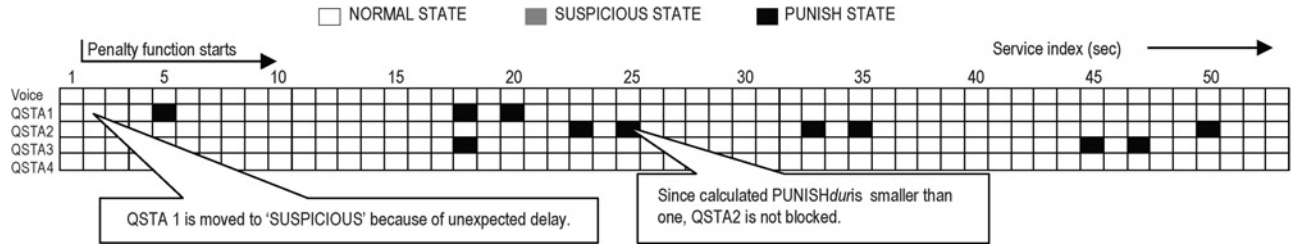


Fig. 7. Result from the penalty function with cumulated random delay without a cheater.

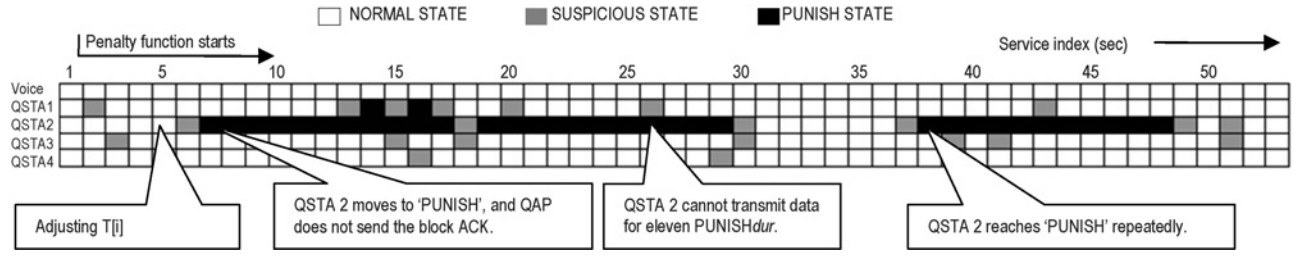


Fig. 8. Result from the penalty function with cumulated random delay.

VI. Simulation

We simulated our proposed mechanism on the NS-2 with an IEEE 802.11e EDCA implementation by TKN. We set a total of five QSTAs. Even though there exist different frame correction mechanisms such as a forward error correction [15], we adopted an automatic repeat request based mechanism, which is the most widely used one in WLANs. Moreover, we implemented the new technical concept block ACK, which has been included in IEEE 802.11n with EDCA. We set the beacon index interval to 1 s to run our penalty function. Since there is no previous observed value of T[i] in (13) at the beginning of the network communication, we initially set 200 ms as T[0] which is the mean value of the on-air delay for each data transmission. When the QAP detects any abnormal situation on the network, it starts our penalty function. To simulate malicious behavior by the cheaters, we set our experimental environment with associated network parameters as follows.

1) Total of five QSTAs on the network. We consider each QSTA has only one multimedia application to get exact results by excluding the possibility of virtual collision among the QoS applications in the same QSTA.
2) The five QSTAs include: a) voice QSTA having the VoIP application with AC_VO, and b) QSTA1, QSTA2, QSTA3, and QSTA4 having the video applications with AC_VI.
3) Values of TXOPLimit are 1.504 ms for AC_VO and 3.008 ms for AC_VI.
4) VoIP and video applications have the same data transmission rate, 1 Mb/s, and same packet size, 8000 bits. The values of these two parameters can be optimized in terms of network throughput [16]. However, our purpose is observing the performance of the proposed solution with the constant distribution of the packet length and generation by assigning the same data generation rate to all QSTAs.
5) In addition, all five QSTAs are located at the same distance from the QAP. As we mentioned, we consider the block ACK for each TXOP as a MAC mechanism. That is, if the QSTA could not successfully receive the ACK frame before the expiration of ACK timer, it assumes there was a collision on the channel, and starts the backoff mechanism.

TABLE III
EDCA Parameter Set Used for Simulation

AC	AIFSN[AC]	TXOPLimit (ms)	CWmin[AC]	CWmax[AC]
AC_VI	2	3.008	CWmin/8	CWmin/4
AC_VO	1	1.504	CWmin/16	CWmin/8

Fig. 5 shows a simulation result without any cheating QSTA for 100 beacon indexes. The QSTA with voice application transmitted approximately 184 kbits/s as a mean value while the other four QSTAs with video applications transmitted approximately 139 kbits/s. This is a reasonable distribution based upon the following fact: even though the voice data has much smaller TXOPLimit than that of the video data, it can transmit data more frequently because it has much smaller values of AIFS, CWmin, and CWmax as shown in Table III.

Conversely, Fig. 6 shows a result with the cheater, QSTA 2, which has maliciously increased its TXOPLimit to three times larger than the other three QSTAs supporting video applications. Thanks to its malicious behavior, QSTA 2 eventually acquires much higher throughput than the other QSTAs. It is even higher than the station with the VoIP application. From the simulation, the voice application was able to transmit 182 kbits/s as a mean value because of its advantaged EDCA parameters. However, the other three QSTAs with Video application, QSTA 1, QSTA 3, and QSTA 4, transmitted much smaller amounts of data. QSTA 1 transmitted 106 kbits/s, QSTA 3 transmitted 76 kbits/s, and QSTA 4 transmitted 87 kbits/s, because QSTA 2 maliciously occupied approximately 42% of total channel bandwidth transmitting 328 kbits/s as a mean value. More importantly, these honest QSTAs could not maintain their stable QoS because they could not finish

TABLE IV
APPROXIMATE MEAN VALUES OF THROUGHPUT (KBITS/S) FOR 100 BEACONS

	Voice	QSTA1	QSTA2	QSTA3	QSTA4
Without cheater	184	142	131	137	146
With cheater (QSTA2)	182	106	328	76	87
After applying penalty function	192	151	146	158	158

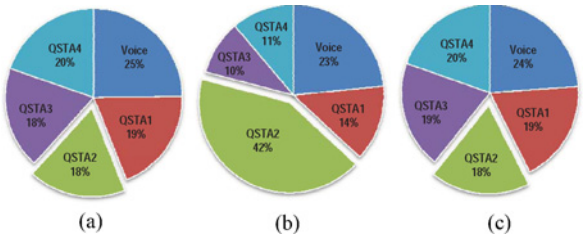
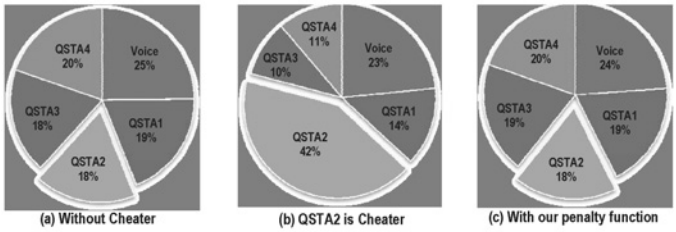


Fig. 9. Pie charts for bandwidth share. (a) Without cheater. (b) QSTA2 is cheater. (c) With our penalty function.



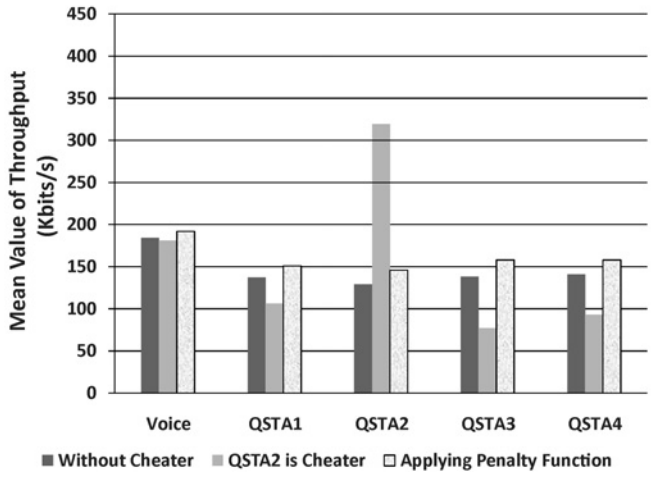


Fig. 10. Mean value of throughput with/without cheater, QSTA2, and after applying our proposed penalty function.

transmitting their data at the minimum required level. The mean values of throughput for all three scenarios are summarized in Table IV.

Fig. 7 shows a time chart representing a result of our penalty function without a cheater. The EDCA network involves multiple delay factors such as transmission delay, propagation delay, processing delay, queuing delay, and transmission collision. Due to these factors, some of data transmission by the honest QSTAs can be identified as a cheater by our penalty function, although this probability is very low, perhaps less than 10% of all observations. In this case, the QAP is able to consider the fact that the QSTA is suffering from delay factors, disturbing the QSTA in such a way to prevent its meeting an assigned TXOPLimit value. In order to handle this case, the punishment decision can be made by referencing the calculated PUNISH*dur*. That is, if the value is smaller than one, then the corresponding QSTAs are not blocked in transmitting their data. However, we need to mention that the most important delay factor is the probability of transmission collision among the QSTAs, because other factors except for queuing delay are relatively constant. Even though the queuing delay has a random property depending on the level of current congestion, we can consider the delay level is also fair over the contenders as long as a proper MAC protocol is enforced.

Fortunately, this probability is uniformly distributed over the QSTAs with the EDCA network. Therefore, this uniform probability distribution allows the imposed penalty to the honest QSTAs at the current beacon index to be sufficiently compensated at the next beacon index for the QSTAs. Note that the QAP starts the penalty function, only when the channel is too crowded causing too many collisions. Moreover, our penalty function does not detect any voice data transmission as a malicious behavior, which is very jitter sensitive, and its packet size is commonly smaller than one from an ordinary video application. As shown in Fig. 7, the QAP can determine that the channel is simply overloaded by honest QSTAs after observing particular beacon index intervals because the PUNISH*dur* values are smaller than one. Therefore, the detected QSTAs are immediately moved back to the suspicious state.

In contrast, Fig. 8 shows a result from our penalty function which detects QSTA 2 as the cheater. In this case, most of detections force QSTA 2 to move to the Punish state after 6 s, and restrict data transmission for PUNISH*dur* to compensate for other honest QSTAs because the AD value of QSTA 2 is now much bigger than threshold T.

Fig. 9 shows pie charts representing the bandwidth distribution over the QSTAs for all three scenarios. As we can see, when we apply our proposed solution, it shows almost the same distribution as the case without a cheater. Fig. 10 shows our final result. When QSTA 2 increases TXOPLimit, other honest QSTAs cannot transmit their QoS data because QSTA 2 acquires too many time slots on the channel. After applying our penalty function, these honest QSTAs can get a reward by having additional transmission opportunities.

VII. CONCLUSION

In the IEEE 802.11e EDCA network, the malicious stations prevent other honest stations from accessing the shared channel. This is a very real and potentially damaging situation for shared channels. In order to detect and then punish such malicious stations we reviewed the literature to show that this problem has not been resolved to date for the most general case which we presented. The final resolution we suggested detects and then punishes for past behavior, after the honest stations have been punished by reduced bandwidth which was

taken by the cheater(s). To detect the potential cheaters, we proposed the potential cheater detection mechanism which enables identification of the stations maliciously increasing their TXOPLimit values. After cumulating the results from the potential cheater detection mechanism, the penalty function runs to detect and block the actual cheaters among the detected potential cheaters. The penalty function has three states, *normal*, *suspicious*, and *punish*. The QAP does not send the ACK packet to the actual cheater when it reaches the punish state. Owing to the punished duration and reduced transmission opportunity for the malicious stations, other honest stations have more transmission opportunity at the next beacon indexes. This unfortunately does not repair the degradation of the QoS for the injured stations, but it does provide a useful benefit in the next cycle for those honest stations. As a result, our proposed mechanism provides a fair transmission opportunity among the contenders. The results from the simulation show our proposed mechanism detects the actual cheater efficiently in real-time. In further work, we will extend our study to include how the number of states and the size of beacon index group affect the detection and identification of the actual cheaters.

REFERENCES

[1] C. Assi, A. Agarwal, and Y. Liu, "Enhanced per-flow admission control and QoS provisioning in IEEE 802.11e wireless LANs," *IEEE Trans. Vehicular Technol.*, vol. 57, no. 2, pp. 1077–1088, Mar. 2008.

[2] S. H. Kim and B.-H. Roh, "Fast detection of distributed global scale network attack symptoms and patterns in high-speed backbone networks," *KSII Trans. Internet Inform. Syst.*, vol. 2, no. 3, pp. 135–149, Jun. 2008.

[3] P. Kyasanur and N. Vaidya, "Detection and handling of MAC layer misbehavior in wireless networks," in *Proc. IEEE DSN*, Jun. 2003, pp. 173–182.

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

[5] M. Cagalj, S. Ganeriwal, I. Aad, and J. Hubaux, "On selfish behavior in CSMA/CA networks," in *Proc. IEEE INFOCOM*, Mar. 2005, pp. 2513–2524.

[6] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Selec. Areas Commun.*, vol. 18, no. 3, pp. 535–547, Mar. 2000.

[7] L. Guang, C. Assi, and A. Benslimane, "Modeling and analysis of predictable random backoff in selfish environments," in *Proc. ACM MSWiM*, Oct. 2006, pp. 86–90.

[8] M. Raya, J. P. Hubaux, and I. Aad, "Domino: A system to detect greedy behavior in IEEE 802.11 hotspots," in *Proc. ACM MobiSys*, Jun. 2004, pp. 84–97.

[9] M. Liu and M. T. Liu, "A power-saving algorithm combing power management and power control for multihop IEEE 802.11 ad hoc networks," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 4, nos. 3–4, pp. 168–173, 2009.

[10] M. H. Ye, C. T. Lau, and A. B. Premkumar, "Traffic scheduling mechanism based on graph theory for power saving mode of IEEE 802.11 distributed coordinator function," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 4, no. 2, pp. 94–84, 2009.

[11] N. Sarma and S. Nandi, "Service differentiation using priority-based MAC protocol in MANETs," *Int. J. Internet Protocol Technol.*, vol. 5, no. 3, pp. 115–131, 2010.

[12] Y. W. Ahn, A. M. K. Cheng, J. Baek, and P. S. Fisher, "Detection and punishment of malicious wireless stations in IEEE 802.11e EDCA network," in *Proc. IEEE Sarnoff Symp.*, Apr. 2010, pp. 1–5.

[13] IEEE Computer Society, *IEEE Std 802.11e. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2005.

[14] D. J. Deng, L. W. Chang, H. W. Wang, D. C. Huang, and Y. M. Huang, "Is RTS/CTS mechanism effective for WLAN," *J. Internet Technol.*, vol. 11, no. 7, pp. 955–964, Dec. 2010.

[15] L. Han, S. Park, S.-S. Kang, and H. P. In, "An adaptive FEC mechanism using crosslayer approach to enhance quality of video transmission over 802.11 WLANs," *KSII Trans. Internet Inform. Syst.*, vol. 4, no. 3, pp. 341–357, Jun. 2010.

[16] X. Ge, C.-X. Wang, Y. Yang, L. Shu, C. Liu, and L. Xiang, "AFSO: An adaptive frame size optimization mechanism for 802.11 networks," *KSII Trans. Internet Inform. Syst.*, vol. 4, no. 3, pp. 205–223, Jun. 2010.

Yong woon Ahn received the B.S. and M.S. degrees in computer science and engineering from the Hankuk University of Foreign Studies, Seoul, Korea, in 2001 and 2003, respectively. He is currently pursuing the Ph.D. degree in computer science with the Department of Computer Science, University of Houston (UH), Houston, TX.

He is a member of the Real-Time Systems Laboratory, UH, and works for VNet as an AJAX Developer. He developed a client/server system for the online multi-player game, Go, with Seloco, Inc., Seoul, in 2003 and 2004. He has recently developed a geographical information system accepting the generic location data for Baker Hughes, Inc., Houston, in 2008. His current research interests include mobile and wireless sensor networking, real-time systems, fault-tolerant computing, ubiquitous computing with embedded devices, and middleware for scalable network environments.

Jinsuk Baek (M'04) received the B.S. and M.S. degrees in computer science and engineering from the Hankuk University of Foreign Studies, Seoul, Korea, in 1996 and 1998, respectively, and the Ph.D. degree in computer science from the University of Houston, Houston, TX, in 2004.

He is currently an Associate Professor of Computer Science with the Department of Computer Science, Winston-Salem State University (WSSU), Winston-Salem, NC. He is the Director of the Network Protocols Group, WSSU. His current research interests include multimedia communications, scalable reliable multicast protocols, wireless sensor networks, mobile computing, and network security.

Dr. Baek has served and is serving as an active member for various technical activities in the computer network and communication field. Currently, he is serving as an Editor of the *KSII TIIS Journal* and as a reviewer for leading journals, including the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON BROADCASTING, *Wiley Wireless Communications and Mobile Computing*, the Wiley International Conference on Communication Systems, *Elsevier Computer Networks*, and the *Hindawi Journal of Computer Systems, Network, and Communications*. He has also served or is serving as a technical committee member of many important international conferences, symposia, and workshops.

Albert Mo Kim Cheng (SM'83–F'11) received the B.A. degree (Hons.) in computer science, graduating Phi Beta Kappa at the age of 19, the M.S. degree in computer science with a minor in electrical engineering at the age of 21, and the Ph.D. degree in computer science at the age of 25, all from the University of Texas, Austin, where he held a GTE Foundation Doctoral Fellowship.

He is currently a Professor of Computer Science with the Department of Computer Science, University of Houston, Houston, TX, where he is the Founding Director of the Real-Time Systems Laboratory. He was a Visiting Professor with the Department of Computer Science, Rice University, Houston, and with the City University of Hong Kong, Kowloon, Hong Kong. He is the author of the popular senior/graduate-level textbook *Real-Time Systems: Scheduling, Analysis, and Verification* (Wiley, 2nd printing with updates, 2005). He is the author and co-author of over 150 refereed publications in leading journals (including the IEEE TRANSACTIONS ON COMPUTERS, the IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, and the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING) and top-tier conferences (including RTSS, RTAS, RTCSA, ICESS, ICPADS, ISLPED, LCN, ICMCS, COMPSAC, AINA, PADL, IPPS, IPDPS, and ICPP) in his areas of research. His current research interests include real-time and embedded systems, cyber-physical systems, power/thermal-aware computing,

formal verification, software engineering, knowledge-based systems, networking, and operating systems.

Dr. Cheng has served as a technical consultant for several organizations, including IBM, Austin, TX, and Shell, Houston. He is an Associate Editor of the IEEE TRANSACTIONS ON COMPUTERS. He has received numerous awards, including the U.S. National Science Foundation Research Initiation Award and the Texas Advanced Research Program Grant. He has been invited to present seminars, tutorials, panel positions, and keynotes at over 80 conferences, universities, and organizations. He is and has been on the technical program committees (including several program chair positions) of over 160 conferences, symposia, workshops, and editorial boards (including the IEEE TRANSACTIONS ON COMPUTERS since 2011 and the IEEE TRANSACTIONS ON SOFTWARE ENGINEERING from 1998 to 2003).

Paul S. Fisher received the B.A. and M.A. degrees in mathematics from the University of Utah, Salt Lake City, and the Ph.D. degree in computer science from Arizona State University, Tempe.

He is currently a R. J. Reynolds Distinguished Professor of Computer Science with the Department of Computer Science, Winston-Salem State University (WSSU), Winston-Salem, NC. He is the Director of the High Performance Computing Group, WSSU. He has written and managed more than 100 proposal efforts for corporations and the Department of Defense involving teams of one to 15 people. He worked as a consultant to the U.S. Army, U.S. Navy, U.S. Air Force, and several companies over the years. In the 1990s, he commercialized a small business innovation research funded effort and built Lightning Strike, a wavelet compression codec, and then sold the company to return to academics. His current research interests include wired/wireless communication protocols, image processing, and pattern recognition.

Minho Jo (M'07) received the B.S. degree in industrial engineering from Chosun University, Gwangju, Korea, and the Ph.D. degree in computer networks from the Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA, in 1994.

He was a Staff Researcher with Samsung Electronics, Seoul, Korea, and was a Professor with the School of Ubiquitous Computing and Systems, Sejong Cyber University, Seoul. He is currently a Brain Korea Professor with the College of Information and Communications, Korea University, Seoul. He is the Executive Director of the Korean Society for Internet Information (KSII) and the Board of Trustees of the Institute of Electronics Engineers of Korea. His current research interests include cognitive radio, wireless sensor networks, radio frequency identification, wireless mesh networks, security in communication networks, machine intelligence in communications, wireless body area networks, ubiquitous, and mobile computing.

Dr. Jo is the Founding Editor-in-Chief and the Chair of the Steering Committee of the *KSII Transactions on Internet and Information Systems*. He serves as an Editor of the IEEE NETWORK. He is an Editor of the *Journal of Wireless Communications and Mobile Computing* and is an Associate Editor of the *Journal of Security and Communication Networks* published by Wiley. He serves as an Associate Editor of the *Journal of Computer Systems, Networks, and Communications* published by Hindawi. He served as the Chairman of the IEEE/ACM WiMax/WiBro Services and QoS Management Symposium, IWCMC, in 2008. He was the TPC Chair of the IEEE Vehicular Technology Conference in 2010 (VTC 2010-Fall). He is the General Chair of the International Ubiquitous Conference and is the Co-Chair of the International Conference on Ubiquitous Convergence Technology. He was on the technical program committees of the IEEE ICC 2008, 2009, IEEE GLOBECOM 2008, 2009, and was the TPC Chair of the CHINACOM 2009 Network and Information Security Symposium.