

# Detection and Prevention of MAC Layer Misbehavior in Ad Hoc Networks

Alvaro A. Cárdenas, Svetlana Radosavac and John S. Baras  
Electrical and Computer Engineering Department  
and the Institute for Systems Research  
The University of Maryland, College Park, MD 20742  
acardena,svetlana,baras@isr.umd.edu

## ABSTRACT

Selfish behavior at the MAC layer can have devastating side effects on the performance of wireless networks, similar to the effects of DoS attacks. In this paper we focus on the prevention and detection of the manipulation of the backoff mechanism by selfish nodes in 802.11. We first propose an algorithm to ensure honest backoffs when at least one, either the receiver or the sender is honest. Then we discuss detection algorithms to deal with the problem of colluding selfish nodes. Although we have focused on the MAC layer of 802.11, our approach is general and can serve as a guideline for the design of any probabilistic distributed MAC protocol.

## Categories and Subject Descriptors

C.2.0 [Computers-Communication Networks]: General—*Security and Protection*

## General Terms

Design, Security, Algorithms

## Keywords

Ad hoc networks, MAC layer, Intrusion detection

## 1. INTRODUCTION

The communication protocols of different layers of an ad hoc network such as the medium access control (MAC) protocol, the routing protocol and the transport protocol, were designed under the assumption that all nodes would obey the given specifications. However when these protocols are implemented in an untrusted environment, nodes can deviate from the protocol specification in order to obtain a given goal, at the expense of honest participants. A selfish user can disobey the rules to access the wireless channel in order to obtain a higher throughput than the other nodes. A selfish user can also change the congestion avoidance parameters of TCP in order to obtain unfair advantage over the rest of the nodes in the network. In limited power devices, certain nodes might refuse to

forward packets in behalf of other sources in order to save battery power. In all these cases, the misbehaving nodes will degrade the performance of the network from the point of view of the honest participants. To fully address these problems, a layered security mechanism should be deployed in order to enforce cooperation or to penalize misbehaving nodes. In this paper we focus on the prevention and detection of unfairness and collision of packets, caused by selfish users at the MAC layer in ad hoc networks.

The MAC layer in a communication network manages a multiaccess link (e.g. a wireless link) so that frames can be sent by each node without constant interference from other nodes. A fairly used MAC protocol for wireless networks is the IEEE 802.11 MAC protocol, which uses a distributed contention resolution mechanism for sharing the wireless channel. Its design tries to ensure a relatively fair access to the medium for all participants of the protocol. In order to avoid collisions, the nodes follow a binary exponential backoff scheme that favors the last winner amongst the contending nodes. One problem with the 802.11's MAC protocol is that even when all contending nodes are well behaved, this mechanism can lead to short time unfairness under the capture effect: nodes that are heavily loaded tend to capture the channel by continuously transmitting data making lightly loaded neighbors to backoff continuously. Very similar effects are obtained when one of the contending nodes is selfish.

MAC layer misbehavior is possible in network access cards that run the MAC protocol in software rather than hardware or firmware, allowing a selfish user or attacker to easily change MAC layer parameters. Even network interface cards implementing most MAC layer functions in hardware and firmware usually provide an expanded set of functionalities which can be exploited to circumvent the limitations imposed by the firmware [2]. In the worst case scenario a vendor might create NIC cards violating the MAC protocol to create an improved performance of its products.

A selfish node in the MAC layer will try to maximize its own throughput and therefore will keep the channel busy. As a side effect of this behavior, regular nodes cannot use the channel for transmission, which leads to a denial of service (DoS) attack [8]. A selfish user can implement a whole range of strategies to maximize its access to the medium. The most likely strategy that a selfish user will employ is to use different schemes for manipulating the rules of the MAC layer. In 802.11, the attacker can manipulate the size of the Network Allocation Vector (NAV) and assign large idle time periods to its neighbors, it can decrease the size of Interframe Spaces (both SIFS and DIFS), it can select small backoff values, it can deauthenticate neighboring nodes etc. A successful detection scheme should take into account all possible cheating options in the MAC layer and detect both: users that employ only one scheme

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SASN'04, October 25, 2004, Washington, DC, USA.  
Copyright 2004 ACM 1-58113-972-1/04/0010 ...\$5.00.

and users that employ a combination of several schemes (e.g. first choosing small backoff values, then assigning large NAV values to its neighbors etc).

One of the most challenging detection tasks is that of detecting backoff manipulation [12, 2]. Due to the randomness introduced in the choice of the backoff, it is difficult to detect when a node has chosen small backoff values by chance or not. In this work we focus on prevention and detection of the manipulation of the backoff mechanism of 802.11's MAC protocol, although our approach can be extended to any probabilistic distributed MAC protocol.

The organization of this paper is the following. The next section summarizes related work. Section 3 presents an introduction to the MAC protocol 802.11 DCF. In section 4 we present an algorithm that prevents cheating in the backoff stage of 802.11 DCF for non-colluding nodes. In Section 5 we present algorithms for detecting misbehavior of colluding nodes. In the last section we discuss future research directions and conclude the paper.

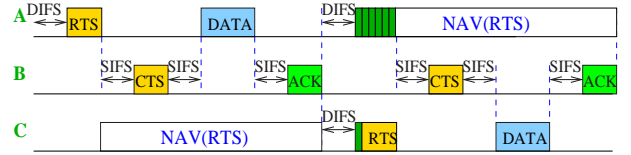
## 2. RELATED WORK

Selfish misbehavior at the MAC layer has been addressed mostly from a game theoretic perspective considering all nodes are selfish. The goal in a game theoretic setting is to design distributed protocols that guarantee for each node, the existence, uniqueness and convergence to a Nash equilibrium with an acceptable throughput. As we have previously pointed out, if users try to maximize their throughput, every node will attempt to transmit continuously in such way that users will deny access to any other node and the network would collapse. This network collapse due to aggressive selfish behavior is a Nash equilibrium. In order to obtain a different Nash equilibrium, each node needs to be assigned a cost for each time it accesses the channel. For example in [11, 1], they consider the case of selfish users in Aloha that attempt to maximize their throughput and minimize the cost for accessing the channel (e.g. energy consumption). Another game theoretic scheme for CSMA/CA schemes is presented in [6]. It shows how a Nash equilibrium is achieved among selfish users when the cost for accessing the channel repeatedly is being jammed by another node. A node jams anonymously any other node that achieves higher throughput than the average of everyone else (assuming nodes always have data to transmit, the throughput of every node should be fair). They assume all nodes are within wireless range in order to avoid the hidden terminal problem, so this scheme is mostly intended for wireless LANs.

Since game theoretic protocols assume all nodes are selfish (the worst case scenario), the throughput achieved in these protocols is substantially less than in protocols where the honest majority cooperates. Under the assumption of an honest majority, detection of misbehaving nodes becomes the primary goal in dealing with misbehavior.

Several possible schemes of node misbehavior in 802.11 for achieving a higher throughput are presented in [12]. The detection of such misbehavior is achieved through a system called DOMINO. However, their detection scheme for backoff manipulation, based on comparing average values of the backoff to given thresholds, is a suboptimal detection technique for every strategy of the greedy user. In section 5 we propose new detection schemes for the backoff manipulation that we believe will improve the performance of systems such as DOMINO.

Kyasanur and Vaidya [10] propose a modification to 802.11 for facilitating the detection of misbehaving nodes. In their scheme, the receiver (a trusted host -e.g. base station-) assigns the backoff value to be used by the sender, so the former can detect any misbehavior of the latter and penalize it by increasing the backoff val-



**Figure 1: Nodes A and C contend for accessing node B. The first time A reserves the channel, and in the second time C accesses the channel.**

ues for the next transmission. The protocol consists of Detection, Penalty and Diagnosis Schemes. The sender is considered to be deviating from the protocol if the observed number of idle slots,  $B_{act}$ , is smaller than a specified fraction  $\alpha$  of the assigned backoff  $B_{exp}$ . For a detected node, a penalty for the next assigned backoff is selected given a measure of the deviation  $D = \max(\alpha B_{exp} - B_{act}, 0)$ . If the sender deviates repeatedly, i.e. if the sum of misbehavior in a sliding window is bigger than some threshold, then the sender is labeled as misbehaving and the receiver takes drastic measures, e.g. drop all packets by the sender.

The problem of applying this protocol for ad hoc networks is that the receiver might not be trusted. In section 4 we extend the idea of [10] by presenting an algorithm that ensures a honest backoff selection among the sender and a receiver as long as one of the participants does not misbehave.

All the schemes presented above as well as the ones we propose, require the proper use of MAC layer authentication schemes, providing uniquely verifiable identities in order to prevent impersonation and Sybil attacks [7].

We also assume that there is a reputation management system similar to CONFIDANT [5, 4], where nodes can monitor and distribute reputation values about other nodes behavior at the MAC layer (CONFIDANT however focuses in reputation at the routing layer). The design of a robust MAC layer reputation system and response is essential and one of our main topics of future work.

## 3. IEEE 802.11 DCF

The distributed coordinating function (DCF) of 802.11 specifies the use of CSMA/CA to reduce packet collisions in the network. A node with a packet to transmit picks a random backoff value  $b$  chosen uniformly from the set  $\{0, 1, \dots, CW - 1\}$  ( $CW$  is the contention window size), and transmits after waiting for  $b$  idle slots. Nodes exchange request to send (RTS) and clear to send (CTS) packets to reserve the channel before transmission. Both the RTS and the CTS contain the proposed duration of data transmission: the duration field indicates the amount of time (in microseconds) after the end of the present frame that the channel will be utilized to complete the successful transmission of the data or management frame. Other hosts which overhear either the RTS or the CTS are required to adjust their network allocation vector (NAV), which indicates for how long should the node defer transmissions on the channel, including the SIFS interval and the acknowledgment frame following the transmitted data frame. If a transmission is unsuccessful (by the lack of CTS or the ACK for the data sent), the  $CW$  value is doubled. If the transmission is successful the host resets its  $CW$  to a minimum value  $CW_{min}$ .

Fig. 1 shows an example of contending nodes using the protocol.

Typical parameter values for the MAC protocol depend on the physical layer that 802.11 uses. For example table 1 shows the parameters used when the physical layer is using direct sequence spread spectrum (DSSS).

DIFS	50 $\mu$ s
SIFS	10 $\mu$ s
SlotTime	20 $\mu$ s
ACK	112bits+PHY_header=203 $\mu$ s
RTS	160bits+PHY_header=207 $\mu$ s
CTS	112bits+PHY_header=203 $\mu$ s
DATA	MAC_header (30b)+DATA(0-2312b)+FCS(4b)
Timeouts	300-350 $\mu$ s
$CW_{min}$	32 time slots
$CW_{max}$	1024 time slots

Table 1: Parameters for DSSS

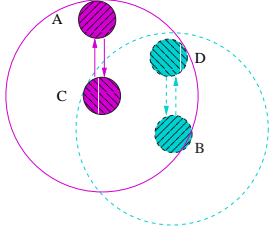


Figure 2: Node C transmits to A and node B wants to transmit to D. After hearing the backoff assigned by A to C, node D assigns a backoff to node B such that it collides with C.

#### 4. ERA-802.11: ENSURING RANDOMNESS FOR 802.11

As we have discussed before [10] requires the receiver to be trusted. This assumption is well suited for infrastructure-based wireless networks, where the base station can be trusted. However, in the case of ad hoc networks the receiver can misbehave by selectively assigning the backoff values to different senders. Depending on the concrete situation, a receiver may benefit by assigning small backoff values to a particular sender (when data from that particular sender needs to be received) or by assigning large backoff values to different neighbors (when it wants to degrade overall performance of neighbors and improve its own throughput). Furthermore, existence of multiple sender-receiver pairs in the interference range of each other creates additional security issues. More specifically, a malicious receiver  $D$  in Fig. 2 can overhear the backoff value assigned to node  $A$  by node  $C$  and unilaterally select a backoff for node  $B$  in order to create a collision with  $C$ .

In this section we propose an extension to the 802.11 CSMA/CA protocol that ensures a uniformly distributed random backoff, when at least one of the parties is honest. The basic idea follows the protocol for flipping coins over the telephone by Blum [3]. The goal is that the sender and the receiver agree through a public discussion on a random value. The main property of the protocol is that an honest party will always be sure that the agreed value is truly random. For an honest sender this means that he can expect a fair treatment in order to access the channel. On the other hand an honest receiver can monitor the behavior of the sender and report a misbehaving node to the reputation management system.

To detect sender deviation from the agreed backoff, the detection algorithm given in [10] can be used. However, the receiver can sense the channel to be busy (channel reserved by a hidden node to the sender) and under normal operation it would not decrement its timer. In order to avoid false positives created by the hidden terminal unit, the receiver always decrements the counter that mon-

itors the sender. It is also mentioned in [10] how to handle the detection during packet retransmissions, i.e. when the sender collides and has to choose by itself another backoff value from the set  $\{0, 1, \dots, (CW_{min} + 1)2^{i-1} - 1\}$  (where  $i$  is the number of retransmissions). Recall that  $CW$  keeps increasing until it reaches  $CW_{max}$ .

The protocol we propose can be embedded in 802.11 and used every time a new reservation of the channel takes place. The messages are appended (denoted by a double bar  $||$ ) to the normal message exchange of 802.11:

$$\begin{array}{l}
 \underline{S} \\
 n \leftarrow \{0, 1\}^{k_{nonce}} \\
 \\
 r' \leftarrow \{0, \dots, CW_{min} - 1\} \\
 \\
 \text{Commit}(r||n) \stackrel{?}{=} \sigma \\
 b_i = r_i \oplus r'_i \\
 \text{for } i \in \{1, \dots, m\}
 \end{array}
 \quad
 \begin{array}{l}
 \underline{R} \\
 r \leftarrow \{0, \dots, CW_{min} - 1\} \\
 \sigma = \text{Commit}(r||n) \\
 \\
 \sigma' = \text{Open} \\
 \\
 b_i = r_i \oplus r'_i \\
 \text{for } i \in \{1, \dots, m\}
 \end{array}$$

We now explain the protocol step by step.

1. In the first step the sender  $S$  selects a nonce: a number  $n$  selected uniformly from the set  $\{0, 1, \dots, 2^{k_{nonce}}\}$ , denoted as  $n \leftarrow \{0, 1\}^{k_{nonce}}$ .  $k_{nonce}$  is a security parameter indicating the level of difficulty of guessing  $n$ . For example  $k_{nonce} = 64$ . This step is optional and is done in order to prevent an offline attack on the commitment scheme.
2. In the second step the receiver  $R$  selects a random backoff  $r$  from the set  $\{0, 1, \dots, CW_{min} - 1\}$  and commits to it. In binary notation  $r$  is a random bit string of length  $m$  ( $r = r_1 r_2 \dots r_m$ ), where  $m = \log_2 CW_{min}$  (note that the contention window size  $CW$  is always a power of two). The commitment scheme Commit is such that the following two properties are satisfied (at least before the time-out for channel reservation:  $300\mu s - 350\mu s$ ):
  - Binding:** After sending  $\text{Commit}(r||n)$ , the receiver cannot open the commitment to a different value  $\tilde{r} \neq r$  (except with negligible probability). This protects against a dishonest  $R$  that might try to change the committed value depending on the  $r'$  received by  $S$ .
  - Hiding:** Given  $\text{Commit}(r||n)$ ,  $S$  cannot extract any information about  $r$  that will enable it to distinguish  $r$  from any other bit string of length  $m$  (except with negligible probability). This protects against a dishonest  $S$  that will try to tailor  $r'$  based on its guess of  $r$ .
3. After receiving the Commitment  $\sigma$ ,  $S$  selects a random value  $r' = r'_1 r'_2 \dots r'_m$  from  $\{0, 1, \dots, CW_{min} - 1\}$ .
4. Finally  $R$  opens its commitment to  $S$ . Opening a commitment is an operation that reveals the committed value  $r$  and some additional information to  $S$ . This enables the other party to verify that the revealed and committed values are the same. If the value opened by the  $R$  is correct, both sender and receiver compute the backoff  $b = b_1 b_2 \dots b_m$  as the xor of the bits:  $b_i = r_i \oplus r'_i$ . Otherwise, the sender can report misbehavior of the node to the reputation management system.

Several commitment schemes are known under very different computational assumptions. Very efficient commitment schemes in terms

of computation and communication, can be implemented under the random oracle model. In this setting it is a standard practice to assume that hash functions  $H$  such as SHA-1 or MD5 are random oracles. Under this assumption it is easy to confirm that the following commitment scheme satisfies the binding (by assuming  $H$  is collision resistant) and hiding properties (by assuming  $H$  is a random oracle):

$$\begin{array}{l} \underline{\text{Commit}}(r||n) \\ i \leftarrow \{0, 1\}^k \\ \text{Output} = H(i||r||n) \\ \\ \underline{\text{Open}} \\ \text{Output} = (i, r) \end{array}$$

where  $k$  is a security parameter (e.g.  $k = 64$ , since it is not considered feasible to search for  $2^{64}$  messages given the current state of art). To open the commitment,  $R$  has to send both  $r$  and  $i$  so that  $S$  can check validity of the commitment.

We now consider 802.11 with Direct Sequence Spread Spectrum (DSSS) physical layer. In DSSS mode the minimum contention window size is 32 time slots, therefore  $m = \log_2 CW_{min} = 5$ , that is,  $r'$  and  $r$  are only 5 bits long which is an insignificant quantity to be appended to a *DATA* frame. The acknowledgement frame is appended  $k + m = 69$  bits.

If we use SHA-1 to implement the hash function of the commitment then we obtain a message digest of 160 bits. The *CTS* frame is doubled in size if the full message digest is used. If doubling the size of a *CTS* frame is a concern, the output of SHA-1 can always be truncated (for example to 80 bits). The security reduction of the message digest has to be evaluated under the birthday paradox: if the message digest has  $h$  bits, then it would take only about  $2^{h/2}$  messages (out of  $2^{k+m+k_{nonce}}$ ), chosen at random, before one would find two (inputs) with the same value (message digest). Considering the normal timeout between frames to be  $300\mu s$ , we can safely assume  $2^{40}$  computations cannot be done in this time.

Finally the nonce parameter should discourage offline attacks. If  $k_{nonce} = 0$ , an attacker could attempt to find collisions offline and then open a false commitment. However, there is a tradeoff then between the length of the message digest used for the commitment and the security parameter  $k_{nonce}$ . Therefore, the probability of a successful offline attack decreases with length of the message digest. If the nonce from the protocol is removed, we can force the sender to use more transmission power than the receiver by requiring the sender to commit the random value instead of the receiver. In this case, the receiver will only be required to append extra five bits to the *CTS* frame.

## 5. DETECTION SYSTEM

The algorithm presented in Section 4 is not resistant to colluding nodes. When sender and receiver collude by selecting their backoff a priori, they can deny access to the network to neighboring nodes. For example, consider Fig. 3 and assume  $C$  is within wireless range of nodes  $D$  and  $M$  (it is a reasonable assumption that in wireless networks there will always be nodes that are neighbors of both colluding nodes:  $D$  and  $M$ ). Without loss of generality, assume  $C$  monitors the access times of node  $M$ . Note that  $C$  can compute the exact backoff value of its neighboring nodes by listening to the exchanged values  $n, \sigma, r', \sigma'$  (between  $M$  and  $D$ ) and then computing the backoff  $b_i = r_i \oplus r'_i$ . If the sender deviates from this backoff then node  $C$  can detect a misbehaving sender in the same way a honest  $D$  would detect a misbehavior of  $M$ . However, if nodes  $D$  and  $M$  collude, they can select their numbers a priori.

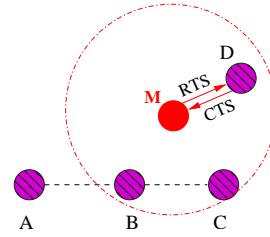


Figure 3: Nodes  $M$  and  $D$  collude and interfere in the communication path of nodes  $B$  and  $C$

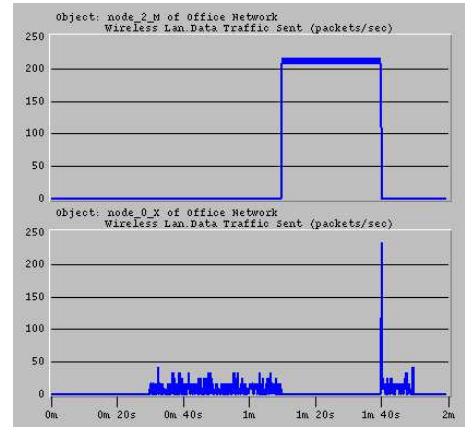


Figure 5: Simulation of traffic sent by node  $M$  (top figure) versus traffic sent by node  $B$  (top figure). When  $D$  and  $M$  collude  $A$  is denied access to the network.

For example they can collude to present a valid message agreement on the backoff zero to node  $C$  by selecting the following values:

$$\begin{array}{ccc} \underline{M} & & \underline{D} \\ n \leftarrow \{0, 1\}^{k_{nonce}} & \xrightarrow{\text{RTS}||n} & \\ & \xleftarrow{\text{CTS}||\sigma} & \sigma = \text{Commit}(00000||n) \\ & \xrightarrow{\text{DATA}||00000} & \sigma' = \text{Open}(\sigma) \\ & \xleftarrow{\text{ACK}||\sigma'} & \\ b = 00000 & & b = 00000 \end{array}$$

In Fig. 4 we show how the sequence of small backoffs  $0, 1, 2, \dots$  from node  $M$  causes the *CTS* timer of node  $A$  to time out. Node  $A$  will therefore repeatedly backoff exponentially, decreasing its chances for accessing the network. This setting was simulated in the network simulator Opnet such that the colluding nodes transmit with no backoff in a time period. Fig. 5 shows how node  $A$  is denied network access while colluding nodes communicate.

Having motivated the need to detect colluding MAC layer misbehavior in ad hoc networks, we now focus on a misbehavior detection mechanism. More specifically, we are interested in designing algorithms for detection of random backoff violations. Although our emphasis is on ad hoc networks, the same algorithms can be applied for monitoring greedy behavior at WLAN access points using the original 802.11 protocol (802.11 without modifications introduced in the previous section).

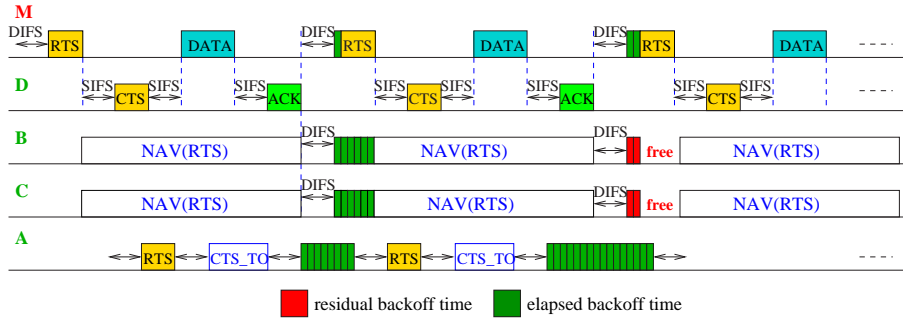


Figure 4: Nodes M and D collude and select a very small backoff, thereby denying access to node A by causing CTS timeouts.

## 5.1 Test for backoff manipulation

We now consider detection strategies in the presence of an intelligent misbehaving node, i.e. a node that is aware of the existence of monitoring neighboring nodes and will adapt its behavior in order to avoid detection. In general, we assume that the colluding nodes are:

1. *knowledgeable*: they know everything a monitoring node knows about the detection scheme.
2. *intelligent*: they can make inferences about the situation in the same way as the monitoring nodes can.

The goal of this class of nodes is to avoid the probability of misbehavior detection ( $P_D$ ) while maximizing their own throughput.

However, it is difficult to come up with a universal strategy that the misbehaving nodes will use for achieving this goal. A naive intrusion detection system (IDS) may assume that the misbehaving nodes will select all their backoff values to be very small. Therefore a model the IDS can assume for the attack is that the misbehaving nodes select their backoff uniformly from the set  $\{0, 1, \dots, CW_{min}/4\}$ . Given this model the IDS would raise an alarm when any of the monitored nodes do not backoff an amount larger than  $CW_{min}/4$ , after  $K$  observations (where  $K$  is chosen given an acceptable false alarm rate). However, an intelligent misbehaving node will easily defeat our detection mechanism by selecting a backoff of zero  $K-1$  times and selecting a value above  $CW_{min}/4$  as the  $K$ th backoff.

### 5.1.1 Tests for change in the mean

The first intuitive assumption to make about the strategy of the colluding nodes is that it will let one of them access the channel in a way that the mean access should decrease from the minimal mean backoff  $B_{min} := (CW_{min} - 1)/2$ , i.e. on average the selfish node will attempt to access the channel more frequently than any other contending node. In order to also penalize nodes that do not double their contention window every time they collide, we could test for a decrease from a nominal backoff  $B_{nom}$  (where  $B_{nom} \geq B_{min}$ ) representing our long term average backoff. Note that each monitoring node has to estimate  $B_{nom}$  online. The selection of testing either a decrease in  $B_{min}$  or  $B_{nom}$  will depend on the risk assessment of the threat provided by the misbehaving nodes. Without loss of generality and in order to be conservative with our detection mechanism we select to test for deviations from  $B_{min}$ .

Let  $X_i$  denote the  $i$ th backoff time for a given monitored node. After measuring  $n$  backoff times for node  $M$ , we end up with the sequence  $X_1, \dots, X_n$ . We assume the IDS makes a decision after  $n$  observations. Using this sequence of data, in DOMINO [12] a detection mechanism is proposed for testing a deviation from the

reference backoff. The algorithm first computes an average  $X_{ac} = \sum_{i=1}^n X_i/n$ , of the observations taken over a given unit of time (e.g. 10s). After that, the averaged value is compared to the reference backoff :

$$X_{ac} < \gamma B_{min}$$

the parameter  $\gamma$  ( $0 < \gamma < 1$ ) controls the rate for false alarms. An optional parameter  $K$  will only flag a false alarm after the statistic  $X_{ac}$  has exceeded  $\gamma B_{min}$ ,  $K$  times. The authors tested the detection scheme against a misbehaving node whose strategy was to select backoff values uniformly from the set  $\{0, 1, \dots, \lfloor CW_{min} \times \delta \rfloor\}$ , where  $\delta$  ( $0 \leq \delta \leq 1$ ) represents the amount of misbehavior ( $\delta = 0$  meaning that the station transmits without backoff and  $\delta = 1$  meaning no misbehavior). The results show that when the misbehaving node increases its throughput three times more the normal value, then the detection mechanism will always catch him while maintaining a false alarm rate below 0.1.

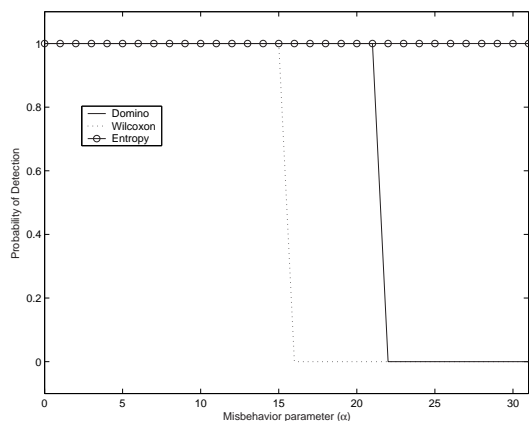
However, this approach is not optimal for more intelligent strategies of misbehaving nodes. A misbehaving node in 802.11 (or a pair of colluding nodes in ERA-802.11) will avoid detection and achieve access to the channel more than half of the contending trials by selecting the following backoff scheme: Select a zero backoff for the first  $(K-1)n$  times (this exploits the fact that an alarm is never raised until the mean statistic has exceeded  $\gamma B_{min}$ ,  $K$  times). After that the optimal strategy to avoid the detection mechanism is to alternate the backoff between zero and  $2\gamma B_{min}$ .

This strategy will ensure that an alarm is never raised for the misbehaving nodes, while still providing access to the channel more than half of the times regardless of the number of contending nodes.

This same fate is shared with several nonparametric tests measuring how “shifted” in the  $X$  axis is the alternative distribution (the attack strategy) from the normal distribution (which is assumed to be symmetric at zero). If we define the independent and identically distributed (i.i.d.) random process:  $Y_1, \dots, Y_n$  by:  $Y_i = B_{min} - X_i$  we can use a Sign Test or a Wilcoxon Test [9].

### 5.1.2 Entropy Estimation

Clearly, if randomness of backoff mechanism of an misbehaving node could be tested, the evasion of detection would be more difficult. However applying empirical statistical tests to random number generators is a highly heuristic affair. Furthermore, an adversary can still cheat several tests, such as cumulative sums, frequency counts and tests based on random walks, while still accessing the channel more frequently than honest participants. We therefore turn our attention to the entropy of the backoff process, since the entropy is one of the most used measures of randomness. Note that we do not need to measure the entropy of the backoff values observed versus the entropy of a uniform random variable



**Figure 6: Entropy Test Limits the Strategy Space of the Misbehaving Nodes**

with range  $\{0, 1, \dots, CW_{min} - 1\}$ . Doing this would cause much higher false alarm rate due to large parameter space for estimation (estimating more parameters means observations before making a decision would increase dramatically). Therefore, we partition the backoff range into  $M$  bins, where  $2 < M < CW_{min}$  so that the estimation of the probabilities in each bin is more efficient for small sample sizes, while limiting the attacker strategies. As a statistical test, we measure the empirical entropy of the  $M$  bins (which ideally should be close to  $-\log_2 \frac{1}{M}$ ) with a threshold given by a fixed false alarm rate.

In order to evaluate the performance of the entropy statistic, we compare it to the detection accuracy of DOMINO and a Wilcoxon nonparametric rank test. The misbehaving strategy is the following sequence of backoffs:  $0, \alpha, 0, \alpha, \dots$ , for  $\alpha \in \{0, \dots, CW_{min} - 1\}$ . Therefore,  $\alpha$  is a measure of evasion by the misbehaving node where  $\alpha = 0$  means the node transmits without any backoff, and  $\alpha = CW_{min} - 1$  means the node alternates between transmitting with no backoff and backing off for  $CW_{min} - 1$ . The parameters of the simulation were  $10^4$  samples,  $n = 20$ ,  $CW_{min} = 32$  and the number of bins to evaluate the entropy was  $M = 8$ . Fig. 6 shows how the entropy test can easily detect the selfishness while the Wilcoxon test misses the strategy when  $\alpha > B_{min}/2$  and DOMINO when  $\alpha > \gamma B_{min}$ . This result should not be a surprise, as the entropy of the selfish behavior in this case is just 2. Note that in the case of  $M = 8$  bins, the optimal strategy for the misbehaving node is to select the smallest backoff within each bin interval: 0,3,7,11,15,19,23,27. Furthermore if the selfish node knows the false alarm rate it can do better. For example when we are only tolerating misbehavior up to an entropy of  $\log_2 6$  with  $M = 8$  bins, then the optimal misbehaving strategy selects the backoff values: 0,3,7,11,15,19,0,3,7,...

## 6. CONCLUSIONS AND FUTURE WORK

Misbehavior at the MAC layer by changing the backoff mechanism can lead to performance degradation and even denial of service attacks in ad hoc networks. In this paper we have presented ERA-802.11 in order to help in the detection of non-colluding selfish nodes. However, even when neighboring nodes know the backoff time agreed by a sender, the network topology, hidden nodes, the exponential backoff due to the capture effect, and network traffic characteristics can severely degrade the correct detection of a misbehaving node. We plan to investigate in future work how the detection accuracy of the monitoring nodes perform with respect

to the number of hidden nodes in their neighborhood and to different types of network traffic. We also plan to investigate how the small overhead included in the reservation packets of ERA-802.11 influences the network throughput.

In the case of colluding nodes, we presented ideas on how intelligent misbehaving nodes can achieve considerable throughput gain while still avoiding usual detection mechanisms. We then presented a new scheme that limits the throughput that an intelligent misbehaving node can obtain if it tries to avoid detection. However for colluding nodes, the problem of detecting backoff manipulation at the MAC layer becomes very difficult. Besides the same detection difficulties that we have for a non-colluding node, we are now required to take several samples of the backoff time in order to come up with an accurate decision. In future work we will focus on a more rigorous treatment of the detection problem and show under the consideration of parameters such as network topology, mobility and traffic characteristics, the difficulty or feasibility of the problem.

Finally, we assumed in this paper that there was a reputation algorithm receiving our detection results. There is still the open question of how to react when we detect a misbehaving node. How bad is the performance degradation for the rest of the network? What is the best punishment strategy? It is our view that the reputation mechanism should have a layered security mechanism in order to provide an educated decision on how to react to MAC layer misbehavior.

## Acknowledgements

We would like to thank Prof. Jonathan Katz for helpful discussions on Coin Flipping over 802.11.

This material is based upon work supported by the U.S. Army Research Office under Award No. DAAD19-01-1-0494 to the University of Maryland College Park.

## 7. REFERENCES

- [1] E. Altman, R. E. Azouzi, and T. Jimenes, "Slotted aloha as a stochastic game with partial information," in *Proceedings of WiOpt*, 2002.
- [2] J. Bellardo and S. Savage, "802.11 denial-of-service attacks: Real vulnerabilities and practical solutions," in *Proceedings of the USENIX Security Symposium*, Washington D.C., August 2003.
- [3] M. Blum, "Coin flipping by telephone: a protocol for solving impossible problems," in *Proceedings of the 24th IEEE Spring Computer Conference, COMPCON*, 1982, pp. 133–137.
- [4] S. Buchegger and J.-Y. Le Boudec, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks," in *Proceedings of Tenth Euromicro PDP (Parallel, Distributed and Network-based Processing)*, Gran Canaria, January 2002, pp. 403–410.
- [5] S. Buchegger and J.-Y. Le Boudec, "Performance analysis of the confidant protocol," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, 2002, pp. 226–236.
- [6] M. Cagalj, S. Ganeriwal, I. Aad, and J.-P. Hubaux, "On cheating in csma/ca ad hoc networks," EPFL, Tech. Rep., February 2004.
- [7] J. R. Douceur, "The Sybil attack," in *Proceedings of the 1st International Peer To Peer Systems Workshop (IPTPS 2002)*, March 2002.
- [8] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of service attacks at the mac layer in wireless ad hoc networks," in *Proc IEEE MILCOM*, October 7-10, 2002.
- [9] J. Hájek, Z. Šidák, and P. Sen, *Theory of rank tests*. Academic Press, New York, 1999.
- [10] P. Kyasanur and N. Vaidya, "Detection and handling of mac layer misbehavior in wireless networks," in *Proceedings of the International Conference on Dependable Systems and Networks*, June 2003.
- [11] A. B. MacKenzie and S. B. Wicker, "Stability of multipacket slotted aloha with selfish users and perfect information," in *Proceedings of the IEEE INFOCOM*, 2003.
- [12] M. Raya, J.-P. Hubaux, and I. Aad, "Domino: A system to detect greedy behavior in ieee 802.11 hotspots," in *Proceedings of the Second International Conference on Mobile Systems, Applications and Services (MobiSys2004)*, Boston, Massachusetts, June 2004.