Telecommun Syst (2013) 52:2397–2406
DOI 10.1007/s11235-011-9552-y

A novel scheme to prevent MAC layer misbehavior in IEEE 802.11 ad hoc networks

Fei Shi · Jaejong Baek · Jooseok Song · Weijie Liu

Published online: 30 July 2011

Abstract The characteristics of ad hoc networks, such as the absence of infrastructure, dynamic topology, shared wireless medium and resource-constrained environment pose various security challenges. Most of the previous research focused on the detection of the misbehavior after it occurred. However, in this paper we propose a new way of thinking to evade the occurrence of misbehavior. In our scheme, Local Most Trustworthy node (LMT node) is allowed to assign the backoff values to originator, rather than permitting the originator to choose the backoff values by itself. With this MAC layer misbehavior avoidance mechanism, the misuse of the backoff in MAC layer in 802.11 DCF can be prevented.

Keywords MAC · Trust management · Misbehavior avoidance · Ad hoc network

1 Introduction

Compared with wired networks and other types of wireless networks, ad hoc networks are more vulnerable to security attacks due to their unique characteristics. Examples of attacks are rushing attack, blackhole attack, MAC layer attack and so on [1, 2]. This paper focuses on misbehavior in the MAC layer by misusing the backoff mechanism.

F. Shi · J. Baek · J. Song (✉)
Department of Computer Science, Yonsei University, Seoul 120-749, South Korea
e-mail: jssong@emerald.yonsei.ac.kr

W. Liu
Department of Information & Industrial Engineering, Yonsei University, Seoul 120-749, South Korea

The MAC layer protocol provided by IEEE 802.11 family [3] of standards was designed for cooperation among nodes in the networks. It assumes that all nodes behave properly and actively. However, attackers could violate the MAC layer protocol simply by misusing the backoff value. Thus, it is essential to develop mechanisms to handle MAC layer misbehavior. In this paper, we propose a scheme to prevent misbehavior in MAC layer in 802.11 DCF. The key idea is to avoid MAC layer misbehavior attack by setting backoff value to originator and monitoring whether originator obeys the backoff.

The organization of this paper follows. In Sect. 2, we describe the model of MAC layer misbehavior attack, and then summarize and discuss the related research. In Sect. 3 we present the details of our scheme. First, we present a trust management mechanism to locate the LMT node. Then we present the attack avoidance mechanism to prevent the misuse in the backoff stage of 802.11 DCF. A formal analysis is presented in Sect. 4. In Sect. 5 we introduce the discussion and future work, and we conclude in Sect. 6.

2 Related work

The Distributed Coordinating Function (DCF) of 802.11 specifies the use of CSMA/CA with the purpose of decreasing collisions in networks. A node which intends to transmit packets picks a random backoff value from [0, CW] (CW is the contention window size), and performs transmission after waiting for backoff value delay. Nodes exchange Request to Send (RTS) and Clear to Send (CTS) packets to reserve the channel before transmission. Other nodes that overhear either the RTS or the CTS are required to defer transmissions on the channel during the conservation period. If a transmission is unsuccessful, the CW value is doubled. If the trans-

mission is successful, the node resets its CW to a minimum value CW_{min}.

However, a misbehaving node may attack MAC layer in several ways. One way is to select backoff values from a different spectrum with smaller average backoff values than the backoff values specified by DCF in 802.11. For example, selecting backoff values from the range $[0, CW/2]$ instead of $[0, CW]$ causes higher possibility of possessing the medium; or selecting backoff values from the range $[CW, 2CW]$ instead of $[0, CW]$ causes the selfish nodes to reduce their consumption and resource; Another way is to use a different retransmission strategy that does not double the CW value after collision which is specified by DCF in 802.11, for example, tripling the CW value or multiplying the CW value by one and half of the CW. The tripling retransmission strategy can make the corresponding node selfish and not participate in the network well. Conversely, the latter one can cause higher opportunity to take up unfair larger bandwidth.

Misbehavior at the MAC layer has been addressed mostly from the game theory. To guarantee the network to reach equilibrium, Cagalj et al. specify the mechanism that each node should follow in terms of controlling channel access probability by adjusting the contention window using a dynamic game model [4]. They present the conditions where the Nash equilibrium of the network with several misbehaving nodes is still Pareto optimal for each node. However, the problem with this scheme is that it assumes that all nodes are within the wireless range, i.e., all nodes can communicate with each other in a direct way. However, this assumption is not satisfied in practical ad hoc networks.

Another way of thinking on the same problem at the MAC layer is provided by Kyasanur et al. [5]. In the paper, the authors propose a modification to IEEE 802.11 to detect misbehaving nodes. In their scheme, the receiver assigns the backoff value to the originator, so that the receiver can detect any misbehavior of the originator. The problem with applying this protocol to ad hoc networks is that the receiver might not be trustworthy, because each node in ad hoc networks works as both a terminal and a router and has equalized security status. It cannot be guaranteed that the receiver is always trustworthy.

3 Proposed scheme

In this paper, we propose a novel scheme to prevent MAC layer misbehavior by avoiding the attacker's misuse of the backoff value in advance. The basic scheme is described as follows. Instead of the originator selecting the backoff values by itself to initialize the backoff counter, in our scheme, the backoff selection is performed by the Local Most Trustworthy node (LMT node), which is one of the originator's neighborhood nodes. Then the originator may use this backoff value as its initial backoff counter for the following transmission. Meanwhile, the LMT node keeps monitoring the originator to observe if it complies with the backoff counter offered by the LMT node. The implementation of the monitoring can be achieved by the watchdog mechanism [6]. By comparing the expected backoff value selected by the LMT node and the actual backoff value used by originator, the LMT node will judge whether the originator node is a MAC layer misbehavior attacker or not.

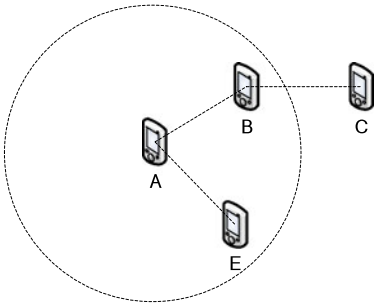


Fig. 1 A scenario of neighbors' connectivity in ad hoc networks

Obviously, two questions are raised. How could we choose the LMT node? How can it be judged if the originator is an attacker or not? We propose two mechanisms to support our scheme for avoiding and detecting MAC layer misbehavior.

3.1 Trust management mechanism

To identify who could be the LMT node, we propose a trust management mechanism that introduces the trust value parameter to evaluate each node's trustworthiness. The trust value ($T = f(C, S)$) is a function of credit and stability values of the nodes. The first component credit value (C) is evaluated based on the transmission behavior of the nodes. The second component stability value (S) indicates if the node moves fast or slowly, or remains stable. In short, the node that exhibits normal behavior and remains stable or moves slowly has the larger trust value. Conversely, the node that performs malicious behavior (i.e., dropping packets) or moves fast has the smaller trust value.

3.1.1 Evaluation of credit value (C)

There are two types of credit value for each node. First, when a node can directly observe another node's behavior, direct credit can be established: $C_d(i, j)$ denotes that there are direct interactions between node i and j. Node i can monitor the behavior of node j and evaluate node j's credit value. As shown in Fig. 1, node A can observe the credit value of node B and E directly and obtain their

direct credit values. Second, when a node receives recommendations from other nodes about another node, recommended credit can be established. There are two types of recommended credits. One type of recommended value is that there is no direct interaction between node i and j, while there may be indirect interactions between them: (1) there is an intermediate node k between nodes i and j; and (2) there are interactions between node i and k and between node j and k. Thus, node i can obtain the credit value of j through k. In Fig. 1, node A and C are not immediate neighbors, but node A can obtain the credit value of C through B's recommendation. Another type of recommended credit value is that there is direct interaction between node i and j, and node i can obtain both the direct value and recommended credit value of node j. In Fig. 1, node A can obtain the direct credit value of E, while node B can recommend the recommended credit value of E to A; in this case, node A will obtain the recommended value of E through B. The recommended credit value is represented by $C_r(i, j)$.

The total credit value $C(i, j)$ can be obtained by integrating $C_d(i, j)$ and $C_r(i, j)$ using the following formula:

$$C(i, j) = \omega_1 C_d(i, j) + \omega_2 C_r(i, j) \tag{1}$$

where ω_1 and ω_2 denote the weight factors for $C_d(i, j)$ and $C_r(i, j)$ respectively. We adopt $\omega_1 > \omega_2$, and $\omega_1 + \omega_2 = 1$. The reason for adopting factor ω_1 larger than ω_2 is that we consider that the direct trustworthiness of one node is more trustful than the recommended trustworthiness from other nodes. Since malicious nodes may provide dishonest recommendation, the recommended credit value should be treated separately from regular direct credit value. Thus, we set the factor ω_2 a relative smaller value to make this part of value less important. Even if there was dishonest recommendation, the damage caused by recommended credit value will be small.

Frequently exchanging recommended credit value between nodes will definitely cause more traffic and the opportunity of transmission collision will be increased. C_r could be recommended only if it is larger than a threshold to decrease the traffic in network and avoid congestion. If not, the recommendation is useless and ignored. In this way, the traffic caused by the recommendation will be decreased, and congestion will be avoided.

To make the issue simple and clear, we make the assumption that the misbehaving nodes only drop packets and they do not modify the content of the packets. It's considered that the nodes in the network should not only share the medium fairly but also carry out their obligations actively. Thus, we consider that the behavior of dropping packets is misbehavior. For example, some nodes dropping packets to cut off the network are malicious nodes and some nodes dropping packets for the purpose of saving their energy are also malicious nodes. The corresponding credit value of the node that drops packets is smaller. The direct credit value (C_d) is established upon observations of whether or not the previous interactions between node i and j are successful. In other words, $C_d(i, j)$ is node i's evaluation of node j by directly monitoring the packet communication of node j. $C_d(i, j)$ can be calculated by node i, using the following formula:

$$C_d(i, j) = \frac{N_j^{act}}{N_j} = \frac{N_j^{out} - N_j^{src}}{N_j^{in} - N_j^{dest}} \tag{2}$$

where the corresponding parameters are interpreted in Table 1.

Table 1 Parameters for evaluating direct credit value (C_d)

Number of packets	Explanation
N_j^{act}	Number of packets actually forwarded by node j
N_j	Number of packets to be forwarded by node j
N_j^{out}	Number of packets that come out from node j
N_j^{src}	Number of packets with node j as the source
N_j^{in}	Number of packets that go into node j
N_j^{dest}	Number of packets with node j as the destination

Equation (2) measures node j's ability to forward packets. Based on packet transmission direction, there are two types of packet related to each node. One type of packet is the packet that "goes into" the node (the number of this type of packet is represented by N_j^{in}); another type of packet is the packet that "comes out" the node (the number of this type of packet is represented by N_j^{out}). Further, the former type of packet ("go into" packet) is divided into two subtypes. One type of "go into" packet is the packet with node j as the destination (the number of this type of packet is represented by N_j^{dest}). Because the destination is node j, this type of packet should not be forwarded. Another type of "go into" packet is the packet that should be forwarded by node j (the number of this type of packet is represented by N_j). So by subtracting the number of packet with node j as the destination (N_j^{dest}) from the number of packet that "goes into" the node (N_j^{in}), the number of packet to be forwarded by node j is obtained (N_j). Furthermore, the type "come out" packet is divided into two subtypes too. One type of "come out" packet is the packet with node j as the source (the number of this type of packet is represented by N_j^{src}). This type of packet is not forwarded but generated by node j. Another type of "come out" packet is the packet that actually forwarded by node j (the number of this type of packet is represented by N_j^{act}). By subtracting the number of packet with node j as the source (N_j^{src}) from the number

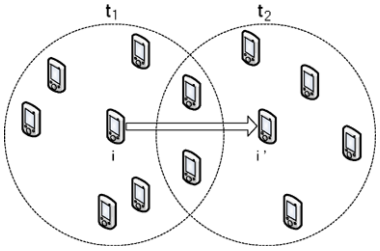


Fig. 2 A scenario showing the change of node i's neighborhood due to the mobility

of packet that "comes out" the node (N_j^{out}), the number of packet that to be forwarded by node j is obtained (N_j^{act}).

3.1.2 Evaluation of stability value (S)

An explanation of the need for the stability value is first called for. Consider the following scenario. One node is moving, and it's so fast that other nodes cannot connect to and communicate with it. In this case, this node is useless and cannot be trusted. In our scheme, we intend to choose the relatively stable LMT node, which may stay in the transmission range of the originator node for a longer time compared to the node that has the high mobility rate. Considering this, the stability value is introduced to weigh up the stabilization of the nodes.

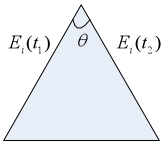
To reasonably describe node stability, this paper uses the graph theory [7] and a similarity computation method [8] to calculate the stability of a node. The network formed by the nodes and the links can be represented by a directed graph, $G(t) = (V, E(t))$, called the neighbor relation graph, wherein $V = \{1, 2, \ldots, N\}$ denotes the number of nodes, and $E(t) = \{e_1, e_2, \ldots, e_m\}$ denotes the number of wireless links. If node i can receive information that is sent from j, there is a directed edge $e(i, j)$ between node i and node j. That is, node j is the neighbor of node i. $E_i(t_1)$ and $E_i(t_2)$ denote the wireless links situation of node i at time point t_1 and t_2 respectively, as shown in Fig. 2. According to the similarity theory, the similarity (S) between $E_i(t_1)$ and $E_i(t_2)$ can be evaluated by the following formula.

$$S_{t_1t_2} = \cos\theta = \frac{E_i(t_1) \cdot E_i(t_2)}{|E_i(t_1)|\,|E_i(t_2)|} \tag{3}$$

where θ is the included angle between the vector $E_i(t_1)$ and $E_i(t_2)$, as the straightforward and intuitive graph shown in Fig. 3.

The $S_{t_1t_2}$ denotes the similarity between node i's neighborhoods status at time point t_1 and t_2. If $S_{t_1t_2}$ is larger, the angle between $E_i(t_1)$ and $E_i(t_2)$, namely θ will be smaller. It expresses more similarity between $E_i(t_1)$ and $E_i(t_2)$, which means that the neighbors of node i do not change dynamically at time t_1 and t_2, namely the node i is relatively stable. On the contrary, if the $S_{t_1t_2}$ is smaller, θ will be larger. It expresses less similarity between $E_i(t_1)$ and $E_i(t_2)$, which means that the neighbors of node i change dynamically at time t_1 and t_2, namely the node i moves fast.

Fig. 3 A intuitive graph representing the similarity between two vectors ($E_i(t_1)$ and $E_i(t_2)$)



3.1.3 Evaluating the trust value (T)

The reward and penalty mechanism is used to evaluate the trust value ($T = f(C, S)$) of each node which depends on the node's credit and stability values. We utilize three mathematic functions to evaluate the trust value of each node. These three functions are the exponential function, the logarithmic function and the logarithmic function.

If $C \geq 0.7$, the logarithmic function is used as follows:

$$T = \log_{(2-C)}(1 + S) \tag{4}$$

If $C \leq 0.3$, the exponential function is used as follows:

$$T = (1 + C)^S \tag{5}$$

If $0.3 < C < 0.7$, the linear function is used as follows:

$$T = C \cdot S \tag{6}$$

First, when the credit value of the node is larger than the threshold, e.g. 0.7, the logarithmic function ($y = \log_a x$) is used to measure its trust value. When value a decreases, y increases with the same x. Logarithmic functions increase quickly with an increase in x, when x is a small number, as exactly the case in our scheme. Therefore, logarithmic functions have fast increasing shapes. In our scheme, logarithmic functions are used to measure the nodes with larger credit values as shown in (4).

Second, when the credit value of the node is smaller than the threshold, e.g. 0.3, the exponential function is used to measure its trust value. Exponential functions ($y = a^x$) are characterized by their growth rate that is proportional to their value. We only use $a > 1$ to describe a node's trust increase in our scheme. Thus, the exponential function $y = a^x (a > 1)$ has a slow increase shape when x is not a large number, and y will increase with an increase in a. Such functions are suitable for measuring nodes with smaller credit value as shown in (5).

Finally, when the credit value of the node is in a medium period, the linear function ($y = a \cdot x$) is used to measure its trust value. With the same value x, function value y increases along with a's increase. Since linear functions have stable increasing shapes, they are used to measure nodes with a stable change in trust or with constantly cooperative

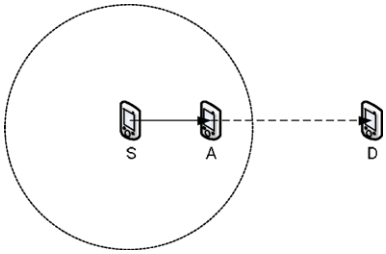


Fig. 4 A scenario of transmission in ad hoc networks

behavior. Thus, linear functions have modestly increasing shapes unlike logarithmic functions and exponential functions. Such functions are suitable for measuring nodes with medium credit value as shown in (6).

In short, if the node has a small credit value, or if it moves faster (stability value is small), its trust value is small. If the node has a large credit value, or if it moves slowly (stability value is large), its trust value is large.

In some case, a malicious node may hide its maliciousness by not moving fast. However, this malicious node will not be assigned as the LMT node due to our trust value evaluation mechanism. Since this malicious node does not move fast, its stability value (S) is relatively large. However, its credit value (C) is relatively smaller than the normal node because of its misbehaviors (i.e., dropping packets). Thus, the trust value (T) of the malicious node is smaller than that of the node which moves slowly and shows good behaviors. It means that this type of malicious node could not be the LMT node in our scheme.

3.1.4 Prediction for faster transmission

The trust management mechanism may cause delay in transmission. In Fig. 4, node S wants to transmit data to node D which is several hops away from it. Node A is the neighbor node of S. According to our trust management scheme, after evaluating the trust value, node A can forward the packet for S, which will cause delay in transmission. To decrease the delay caused by trust value calculation, we propose a prediction scheme to perform some tradeoff between security and speed of transfer.

A node's historical trust record is introduced as a significant factor in measuring its current trustworthiness. Based on the practical circumstance in ad hoc networks, we make the following assumptions that misbehaving nodes are the minority; normal nodes are the majority. The previous trust value is used to predict whether the node is currently a misbehaving node or not without calculating the current trust value before the transmission. A node's current trust value (T_c) is predicted based on its previous trust value (T_p).

As shown in Fig. 4, when node S want to transfer data to D, node A will first check if it has the record of the originator S or not. If there is no record, node A will merely

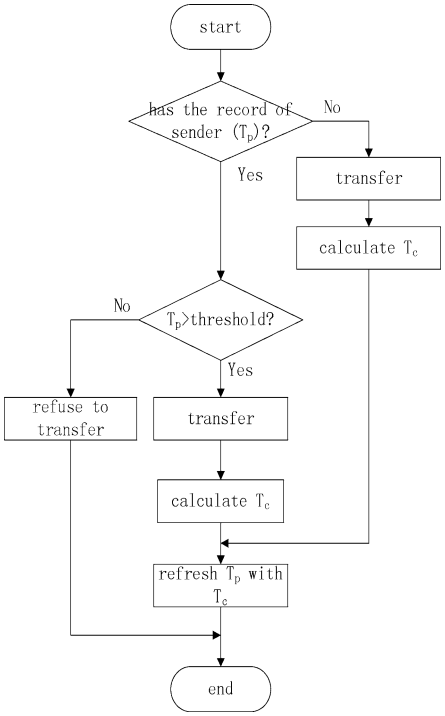


Fig. 5 Flow chart of prediction mechanism

forwards the packet and then calculate the trust value of the originator S. If there is the record of S in A, the adequacy of previous trust value (T_p) will be checked. If its size is inadequate, node A will refuse to forward the packet from node S. If it is larger than threshold, the transmission action will be done first. Then the calculation of the current trust value (T_c) of node S will be evaluated. Finally, the current trust value will be used to refresh the trust value of node S.

In this prediction mechanism, the transmission action is done prior to the trust value calculation action to reduce the delay as shown in Fig. 5. It is allowed to happen for several reasons. First, if there is no record (T_p) of the originator, based on our assumption that normal nodes are the majority, it is predicted that the originator is the normal node. Even if the originator is a misbehaving node, after calculating of the T_c and refreshing the T_p with T_c, the improper prediction will be solved after a period. Second, if there is record of the originator and its T_p is larger than threshold, which means the node was a good node in the past, the same should be believed currently. Thus, the node is predicted to be the normal node and the transmission action is done in the first place to decrease the delay. Even if a good node in the past unfortunately becomes a bad node, after the calculation of the current trust value and the refreshment of the trust value, the improper prediction will be solved. By this prediction mechanism, the security is guaranteed and at the same time the efficiency is increased dramatically.

3.2 Preventing MAC layer misbehavior

In this section, we use two mechanisms to prevent the MAC layer misbehavior: MAC layer misbehavior avoidance mechanism and detection mechanism.

3.2.1 Avoiding MAC layer misbehavior

First of all, an explanation of how to choose the LMT node follows. Using the aforementioned trust management mechanism, the neighbor nodes' trust value in each neighborhood can be obtained. In our scheme, each node maintains a table containing its neighbor nodes' trust values. The table is refreshed based on neighbors' behaviors and stability. For example, the trust value is decreased if the corresponding node performs bad action (dropping packets as we assumed), or the item of a node's trust value is deleted from the trust table if the corresponding node goes out of the transmission range. It is required that the originator node that intends to initialize the data transmission should assign the LMT node by searching through its trust table and choosing the node that has the largest trust value in the trust table.

The following is the process of our MAC layer misbehavior avoidance mechanism.

(1) First, the originator broadcasts the RTS to request the channel, which is the same with the specification of 802.11 DCF. The difference is that the RTS piggybacks the ID of the LMT node. All the neighbor nodes of the originator node including the LMT node will receive the RTS;
(2) After receiving RTS from the originator node, the LMT node replies the CTS piggybacking the backoff value chosen from the range $[0, \mathrm{CW}_{min}]$. CW_{min} is the minimum contention window value used in IEEE 802.11;
(3) Considering the CTS might be lost due to some reasons (e.g. bad channel condition), the LMT node sets a timeout value and observes the behavior of the originator during the period. If the originator does not send any packet during the period, timeout occurs. The LMT node will send the CTS again with the backoff value from the range $[0, 2\mathrm{CW}_{min}]$;
(4) The originator receives this CTS message and check whether the source of the CTS is the LMT node or not. If it is, then the originator extracts the backoff value from it. Then the originator may use this backoff value as its initial backoff counter for the following transmission.

There could be another potential issue that LMT node may probably consume much more battery power in the trust management mechanism. Since the LMT node perform the supererogatory backoff assignment task, its battery may be consumed faster than other nodes. Fortunately, as to different originators, their corresponding LMT nodes will probably change as shown in Fig. 6. In local neighborhood 1, node A's corresponding LMT node is node B. But as to node C and E, since their neighborhoods change (neighborhood 2 and 3), their corresponding LMT node will probably change to other nodes (i.e., node D and F are the corresponding LMT nodes to node C and E respectively in Fig. 6).

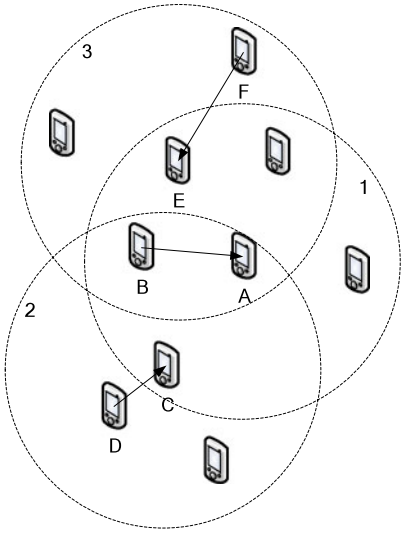


Fig. 6 A scenario of originators and their corresponding LMT node

Table 2 Parameters of detecting MAC layer misbehavior mechanism

Parameter	Explanation
b_{exp}	Expected backoff size set by LMT node
b_{act}	Actual backoff size s used by originator
b_{diff}	Difference between b_{exp} and b_{act}
α, β	Threshold ($\alpha, \beta > 0$)
n	Transmission times of originator
i	ith time transmission of originator

The aforementioned phase is meant to avoid MAC layer misbehavior: LMT node is allowed to set the backoff value for the originator, rather than the originator choosing the backoff value itself. The next phase is meant to let the LMT node continue to monitor the originator.

3.2.2 Detecting MAC layer misbehavior

Besides assigning the backoff value to the originator, the LMT node also keeps monitoring the originator to observe if it complies with the backoff counter offered by the LMT node. The implementation of the monitoring can be achieved by the watchdog mechanism. By comparing the expected backoff value selected by the LMT node and the actual backoff value used by originator, the LMT node will judge whether the originator node is a normal node, or a MAC layer misbehaving node, or a selfish node that does not actively participate in the ad hoc network.

An explanation will first be given of the need for the MAC layer misbehavior detection phase is necessary. Although in our scheme, LMT node is allowed to set the backoff value to the originator, this does not mean that the originator must strictly use this backoff value. This backoff value is only a reference value. The size of actual backoff value should be based on the channel condition at that moment. For example, if the channel condition is good, i.e., the medium is not so busy, the originator could do the routing or transmission actions faster, i.e., choose a smaller backoff value; but if the channel condition is bad, to reduce the collision, the originator could choose a larger backoff value by itself.

Considering all these things, the MAC layer misbehavior detection phase to monitor and evaluate the relationship between the expected backoff (b_{exp}) and the actual backoff (b_{act}). By comparing these two backoff sizes, a final judgment can be made as to whether the originator is a attacker who possesses the bandwidth unfairly, or a selfish node that does not actively participate in the ad hoc networks, and it can be determined if the node is a normal node.

The difference between expected backoff and actual backoff can be calculated by the following formula (Parameters' explanations are in Table 2).

$$b_{diff_i} = b_{exp_i} - b_{act_i}, \tag{7}$$

If

$$\frac{\sum_{i=1}^{n} b_{diff_i}}{\sum_{i=1}^{n} b_{exp_i}} > \alpha \tag{8}$$

the node is a MAC layer misbehaving node, which takes up the bandwidth unfairly and makes the medium busy to other normal nodes;

$$\text{If} \quad \frac{\sum_{i=1}^{n} b_{diff_i}}{\sum_{i=1}^{n} b_{exp_i}} < -\beta \tag{9}$$

the node is a selfish node, which does not participate in the networks actively for saving its resource, e.g., battery power;

$$\text{If} \quad -\beta \leq \frac{\sum_{i=1}^{n} b_{diff_i}}{\sum_{i=1}^{n} b_{exp_i}} \leq \alpha \tag{10}$$

the node is a normal node.

The reference factors of both α and β depend on the channel condition. If the channel condition is good, α is set as a larger value, e.g., 0.5, and β is set as a smaller value, e.g., 0.2. The reason is that when channel condition is good, originator may choose a smaller backoff to decrease the transmission delay. But if channel condition is bad, α is set as a smaller value, e.g. 0.2, and β is set as a smaller value, e.g., 0.5. The reason is that when channel condition is bad, originator may choose a larger backoff to decrease the collision.

4 Analysis of the scheme

In this section, a formal analysis of our scheme is presented, and it is proven that MAC layer misbehavior can be prevented using such scheme. It will be proven that our scheme is not only feasible, but is also efficient and secure. The methodology [9] that is used in this analysis focuses on the trust management mechanism and the MAC layer misbehavior avoidance mechanism, which are the main components of our scheme.

To analyze our scheme, we provide several theorems. Necessary lemmas are proven in advance for the theorems.

Lemma 1 *Any node can classify its neighbors based on their past normal behavior and misbehavior.*

Proof First, any node can observe its neighbors' behavior. Though the watchdog mechanism, which is a method of detecting misbehaving nodes in ad hoc networks, the behavior of neighbors can be observed and monitored.

Second, in our trust management mechanism, a node's behavior is measured by the credit value. The credit degree can be evaluated based on the past behavior record of neighbors using (2). If the credit value is big, the corresponding node had normal behaviors in the past. Conversely, if the credit value is small, the corresponding node exhibited misbehaviors. □

Lemma 2 *Any node can classify its neighbors based on their speed of mobility.*

Proof First, mobility can be inspected by probing message or advertisement mechanism [10, 13]. For example, in AODV [11, 12] routing protocol, hello message indicates the presence of a neighbor. When a node receives a hello message from its neighbor, it creates or refreshes the routing table entry. If a node has not sent any broadcast control message within a specified interval, a hello message is locally broadcast to maintain connectivity. This results in at least one hello message transmission in each time period. Failure to receive any hello message from a neighbor for several time intervals indicates that the neighbor is no longer within the transmission range, and connectivity with it has been lost.

Second, the mobility of the node is measured by the stability value in our trust management mechanism. The stability can be evaluated based on the similarity between two neighborhoods at two time points using (3). If the stability value is big, the corresponding node is stable or moves slowly. Conversely, if the stability value is small, the corresponding node moves fast. □

Lemma 3 *The trust value can be updated for nodes based on their past behavior and mobility.*

Proof The credit value can be obtained from (2). Those parameters (number of different types of packets) vary with the transmission times. When the nodes generate a new transmission, these parameters may change, which induces the updating of the credit value.

The stability value can be obtained from (3). With the passing of time, the neighborhood of any node may change. Thus the stability value will be updated.

The update of the credit and stability values will bring in the update of the trust value according to (4), (5), or (6). □

Theorem 1 *Among the neighborhood nodes of any node, the LMT node can always be chosen.*

Proof of Theorem 1 According to Lemma 1 and 2, it is possible to monitor neighbors' behaviors and mobility. Then the trust value can be obtained using (4), (5), or (6) from the credit and stability values, which can be evaluated using (2) and (3), respectively. According to Lemma 3, the trust value can be updated in a timely manner. Thus, for each node, the neighbors' trust management can be guaranteed, and any node can determine which node in its neighborhood has the largest trust value, namely, the LMT node. □

Lemma 4 *Our trust management mechanism is effective.*

Proof In our trust management mechanism, a trust prediction method is used to evaluate a node's trustworthiness. With this prediction method, the delay due to the trust value calculation disappears. As shown in Fig. 5, this prediction job is implemented prior to the trust value calculation, which increases the efficiency of the trust management mechanism. □

Lemma 5 *Our trust management mechanism can guarantee the security of transmission by using the trust value.*

Proof In our trust management mechanism, each node is identified by a trust value ($T = f(C, S)$). A node's trustworthiness depends on its trust value. If the trust value is larger than the threshold, the node will be identified as a normal node; and if the trust value is smaller than the threshold, it will be adjudged a malicious node.

To lessen delay due to trust management, a prediction mechanism is proposed. Although this prediction mechanism is a tradeoff between security and transmission speed, it is proven to still be secure enough for transmission. As shown in Fig. 5, the transmission is prior to the trust evaluation, which induces fast transmission and prevents delay due to trust evaluation. Because the previous trust value of a node is used to represent the node's current trust situation, this prediction can be implemented to guarantee both speed and security. Based on a node's previous trust value, its treatment can be judged. Even if a node unfortunately suddenly turns from a good node to a bad node or the reverse, after a period of the accumulated calculation of new current trust value, the wrong prediction will be fixed. □

Theorem 2 *Our trust management mechanism can not only guarantee the transmission, judging by the trust value assignment, but can also ensure efficiency and prevent delay in transmission.*

Proof of Theorem 2 According to Lemma 4 and 5, the efficiency and security of our trust management mechanism can both be guaranteed. □

Lemma 6 *If the originators always choose the smaller backoff, which violates randomness in some spectra, this will be detected by the LMT node.*

Proof Based on the (8), if the originator always chooses smaller backoff values, it is the attacker who possesses the wireless bandwidth unfairly and may cause the medium always busy to other neighbors. □

Lemma 7 *If originators always choose larger backoff values, which violate randomness in some spectra, it will be detected by the LMT node.*

Proof Based on the (9), if an originator always chooses larger backoff values, it is the selfish node that does not participate in the networks actively to save the resource, e.g., battery. □

Theorem 3 *The LMT node can judge which originators are exhibiting MAC layer misbehaviors.*

Proof of Theorem 3 Through Lemma 6 and 7, it is known that the LMT node can monitor the originator. LMT node can also detect whether or not originator performs MAC layer misbehavior, namely, refuses to participate in the network actively by always choosing a big backoff value, or unfairly possesses the wireless bandwidth by always choosing a small backoff value. □

5 Discussion and future work

In this section, we discuss the pros and cons of our scheme compared with other works. Then we address the direction of our future work

As mentioned in Sect. 2, one of the major defects of using game theory to prevent misbehavior is that game theory protocols assume that all nodes are selfish, which is different from objective reality of ad hoc networks. On the contrary, our scheme is based on the assumption that the

majority is good-behavior node, and the minority is misbehaving node. Under this assumption, we focus on misbehaving nodes. Another issue of some protocols based on game theory is that they assume that all nodes are within the wireless range, which is not satisfied in practical ad hoc networks. As shown in Fig. 6, in our scheme the LMT node may be in different neighborhood which provides the security for the corresponding originator node in each neighborhood. The additional benefit is that because of the distribution of LMT node in the whole network, the resource consumption is reduced with regards to each LMT node.

The key issue in [5] is that the receiver assigns the backoff values to the originator node, which means that the receiver must be a trustworthy node. However, in ad hoc networks each node has equalized security status. It cannot be guaranteed that the receiver is always trustworthy. Thus, our scheme utilizes and expands this idea by letting LMT node to assign the backoff values to the originator instead of the receiver. The benefit is that the node that assigns the backoff values is trustworthy and the assigned backoff value is dependable.

However, in our scheme there is overhead to calculate the trust value.

(1) For the trust management mechanism, when there is no transmission in the initial time, it is impossible to obtain the number of packets actually forwarded and the number of packets expected to be forwarded. Thus, the credit value cannot be evaluated in the initial time. It can be evaluated only after a period of transmission, which causes delay of the evaluation of trust value.
(2) When a node moves to a new neighborhood, namely, the node is a stranger to its neighbors; its credit value must be recalculated. Then the problem mentioned above may arise again. If the node moves frequently, this problem may become more serious. However, the stability value is a variable of the trust value function in our scheme. If the node moves frequently, its stability value will be small as shown in (3). The node's trust value will be small accordingly, which means that the node will not become the LMT node which has the largest trust value in the neighborhood.

Our future works to enhance our scheme are as follows:

(1) Enhance the trust management mechanism to reduce the delay of determining the LMT node for each originator;
(2) Address the reaction mechanism of LMT node after detecting the misbehavior in MAC layer.

6 Conclusion

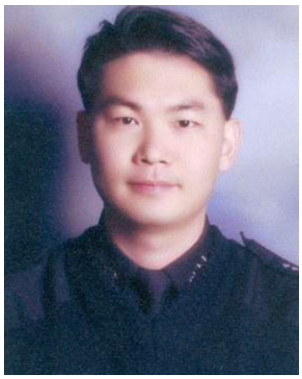
This paper presents initial work in preventing misbehavior in MAC layer caused by misusing of backoff value in 802.11 DCF. Instead of implementing the prevention by detecting the misuse of backoff value, an avoidance mechanism is used to prevent this type of attack. In our scheme, LMT node is chosen to assign the backoff value to the originator. LMT node monitors the actual backoff value used by originator to judge whether the originator is the misbehaving node or not. Finally, according to our analysis of the scheme, the scheme is proven to be feasible and effective.

Acknowledgements This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0016161).

References

1. Kannhavong, B., Nakayama, H., Nemoto, Y., Kato, N., & Jamalipour, A. (2007). A survey of routing attacks in mobile ad hoc networks. *IEEE Wireless Communications*, *14*(5), 85–91.
2. Wang, D., Hu, M., & Zhi, H. (2008). A survey of secure routing in ad hoc networks. In *Web-age information management* (pp. 482–486).
3. IEEE 802.11 Standard for wireless LAN. (1999). *Medium access control (MAC) and physical layer (PHY) specification*. IEEE Inc.
4. Cagalj, M., Ganeriwal, S., Aad, I., & Hubaux, J. P. (2004). *On cheating in CSMA/CA ad hoc networks*. EPFL, Tech. rep., 1–16.
5. Kyasanur, P., & Vaidya, N. (2003). Detection and handling of MAC layer misbehavior in wireless networks. In *International conference on dependable systems and networks* (pp. 173–182).
6. Sergio, M., Giuli, T. J., Kevin, L., & Mary, B. (2000). Mitigating routing misbehavior in mobile ad hoc networks. In *International conference on mobile computing and networking* (pp. 255–265).
7. Faragó, A. (2002). Scalable analysis and design of ad hoc networks via random graph theory. In *Workshop on discrete algorithms and methods for MOBILE computing and communications* (pp. 43–50).
8. Tsumoto, S., & Hirano, S. (2003). Visualization of rule's similarity using multidimensional scaling. In *Data mining* (pp. 339–346).
9. Li, G., Needham, R., & Yahalom, R. (1990). Reasoning about belief in cryptographic protocols. In *Proc. IEEE symposium on security and privacy* (pp. 234–248).
10. Giruka, V. C., & Singhal, M. (2005). Hello protocols for ad-hoc networks: overhead and accuracy tradeoff. In *World of wireless mobile and multimedia networks* (pp. 13–16).
11. Perkins, C. E., & Royer, E. M. (1999). Ad-hoc on-demand distance vector routing. In *Mobile computing systems and applications* (pp. 90–100).
12. Abusalah, L., Khokhar, A., & Guizani, M. (2008). A survey of secure mobile ad hoc routing protocols. *IEEE Communications Surveys and Tutorials*, *10*(4), 78–93.
13. Krco, S., & Dupcinov, M. (2003). Improved neighbor detection algorithm for AODV routing protocol. *IEEE Communications Letters*, *7*(12), 584–586.

Fei Shi received the B.S. degree in computer science and M.S. degree in computer system from the Northeastern University, Shenyang, China, in 2004, and 2008, respectively. He is currently working toward the Ph.D. degree in computer science at Yonsei University, Seoul, Korea. His research interests include wireless communications, mobile ad hoc networks, vehicular ad hoc networks and information security.



Jaejong Baek received his B.S. degree in computer science from Hanbat National University of Technology in 1996, and M.S. degree in network and security from Yonsei University, Seoul, Korea, in 2001. Since September, 2007. He has studied for the Ph.D. in Yonsei University. His research interests include wireless communication, mobility management, cyber warfare and information security.

Jooseok Song received the B.S. degree in Electrical Engineering from Seoul National University, Seoul, Korea, in 1976, and the M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology, Daejeon, Korea, in 1979. In 1988, he received the Ph.D. degree in Computer Science from University of California at Berkeley. From 1988 to 1989, he was an Assistant Professor at the Naval Postgraduate School, Monterey, CA. He was the president of Korea Institute of Information Security and Cryptology in 2006. He is currently a Professor of computer science at Yonsei University, Seoul. His research interests include cryptography and network security.

Weijie Liu received the B.S. degree in computer science and M.S. degree in history of science & technology from the China University of Geosciences, Wuhan, China, in 2006, and 2008, respectively. She is currently working toward the Ph.D. degree in the department of information & industrial engineering at Yonsei University, Seoul, Korea. Her research interests include network security, semantic network and semantic technology in wireless communication, artificial intelligence in wireless communication.