An Efficient Reliable Multicasting Protocol for Micro-cellular Wireless Networks [1]

Dongkyun Kim†, Jaewoo Park, and Yanghee Choi†

School of Computer Science and Engineering†
Seoul National University, San 56-1, Shillim-dong, Kwanak-ku, Seoul, Korea

E-mail : dkkim@jade.snu.ac.kr, {jwpark, yhchoi}@mmlab.snu.ac.kr

Abstract

A reliable multicasting protocol needs to perform the efficient group communication in the micro-cellular wireless networks. In particular, the CTS(or ACK) explosion from the group receivers should be addressed so as to provide all members with the reliability of the data transmitted. Recently, some reliable multicast protocols have been devised to solve the explosion problem : Leader-Based Protocol(LBP) and Random Leader-Based Protocol(RLBP). However, LBP experiences the overhead of maintaining a leader in a cell. While this kind of overhead can be overcome by RLBP utilizing the leader randomly selected during the group communication, RLBP has its disadvantage of the possibility of the occurrence of multiple leaders in the cell. In this paper, we modify the basic RLBP to avoid the occurrence of multiple leaders as well as to overcome the overhead of leader election. Simulation results enables our proposed protocol to be used for an efficient reliable multicasting protocol in the networks.

Keywords *: Wireless LAN, Reliable Multicast, RTS-CTS handshaking, Dynamic Leader Election*

I. Introduction

In addition to wired networks, many research activities on the multicasting mechanism in wireless networks have been performed due to the capability of efficient use for the system resources. Specially, as for wireless networks due to the strict requirement for a wireless bandwidth utilization, an efficient multicast protocol should be devised. In this work, we handle a single channel multi-access wireless LAN consisting of several small cells. For instance of the network, IEEE 802.11 based wireless LAN [3] is currently deployed in many local area network environment.

In this sort of network, two or more packets at the same time sent to the intended receiver cause the packet collision. The successful packet transmission requires only one node to send its data at a time. Hence, two important issues should be addressed[1,2]. First, the channel should be acquired before multicast transmission is initiated. Second, a packet should also be sent to the receivers reliably due to the high error rate over the wireless links. The reliability can be achieved with ARQ-based approaches between the base staton and each receiver. While the channel acquisition mechanism which gets a positive feedback from the intended receiver after sending a request is used efficiently for unicast communication, it is not easy to extend this to a multicast transmission. It is because the feedback packets from all members can collide at the base station. When a sender expects feedback from the receivers for reliable multicast transmission, positive acknowledgments(ACKs) or negative acknowledgments(NAKs) from several group members can also collide at the sender, resulting in delaying any error recovery and wasting bandwidth. In this work, a new efficient approach is presented to overcome this kind of feedback collision problem in single channel multi-access wireless LANs, both for channel acquisition and reliability.

This work considers the microcell-based wireless LAN environment, where a base station located at the center of the cell administers each microcell(Figure 1). In this microcell environment, all terminals in a cell are within the range of one another and the base station. We assume that the nodes in different cells don't interfere each other by using separate CDMA codes. Also, this network allows us not to worry about the hidden terminal problem. Our proposed protocol can be extended enough to solve the hidden terminal problem by using the similar mechanism mentioned in [1]. Additionally, we assume that for the communication between two nodes, they can directly exchange their data packets without the relaying performed by a base station, while all multicast communication should be directed from the base station to the receivers.

Besides, time is assumed to grow in the unit of "slot" and the perfect synchronization, therefore, is assumed to be

[1] This work was supported in part by the Brain Korea 21 project of Ministry of Education, in part by the National Research Laboratory project of Ministry of Science and Technology, and in part by Agency of Defense Development, 2001, Korea.

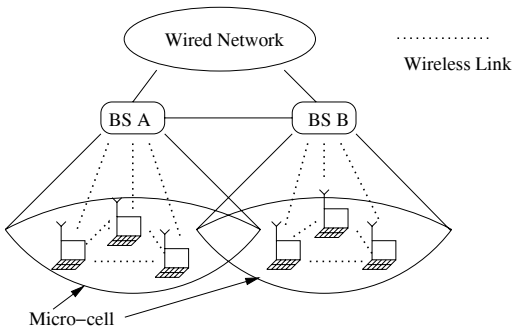

Fig. 1. Micro Cell Environment.

acheived by other synchronization protocols. In a multi-access wireless LAN, the contention to the wireless media before data transmission is performed along with the possibility of the collision. The existing wireless LAN like IEEE 802.11 standard utilizes the handshaking procedure of RTS and CTS messages. This style of RTS-CTS exchange is used for two purposes. One is to avoid the hidden terminal problem, and the other is to prevent the loss of bandwidth caused by collision detection which utilizes ACK or NAK after the entire packet has been transmitted. We now describe the procedure of RTS-CTS exchange for the unicast transmission. When a node wants to send the data packet, an RTS message is sent to the intended receiver. On hearing the RTS, the receiver replies with a CTS. The initiator can start the data packet when it receives the CTS safely. Other nodes overhearing an RTS and a CTS defer their transmission for a duration sufficient for the associated CTS and oncoming data transmission to be sent, respectively.

It is agreed that the RTS-CTS control structure be retained when multicast functionality is overlaid. Consequently, some ways are considered to extend the access control mechanism instead of modifying its basic structure in order to support multicast functionality[1,2]. While the RTS-CTS mechanism can be efficiently used for unicast tranmission, some problems should be addressed in the context of multicasting. The CTS messages from members of the multicast group is likey to collide at the base station. A similar collision problem can also be expected with the repsect to the feedback(ACK or NAK) provided by link-level ARQ mechanism.

In particular, the CTS(or ACK) explosion from the group receivers should be addressed so as to provide all members with the reliability of the data transmitted. Recently, some reliable multicast protocols have been devised to solve the explosion problem, namely, Delayed feedback-Based Protocol(DBP), Leader-Based Protocol(LBP) and Random Leader-Based Protocol(RLBP). LBP experiences the overhead of maintaining a leader in a cell while LBP outperforms DBP utilizing simple delayed-timers among nodes. While this kind of overhead can be overcome by RLBP utilizing the leader randomly selected during the group communication, RLBP has its disadvantage of the possibility of the occurrence of multiple leaders in the cell. In this paper, we modify the basic RLBP to avoid the occurrence of multiple leaders as well as to overcome the overhead of leader election.

This paper is organized as follows. In Section 2, three related works (DBP, LBP and RLBP) are presented. We propose our enhanced protocol in Section 3 followed by performance evaluation in Section 4. Finally, we close this paper with some concluding remarks in Section 5.

II. Related Works

We introduce three protocols proposed by other research groups, namely, Delayed feedback-based(DBP), Leader-based(LBP) and Random leader-based protocols(RLBP). They all assume that a single sender(base station) sends data packets reliably to a group of receivers in a cell. In [1], a fixed leader sends feedback to the base station, resulting in avoiding CTS collision from a group of receivers as well as ACK collision. Also, since the leader-based protocol has the high overhead of maintaining the leader in [1], a leader in [2] can be selected randomly by utilizing Delayed Feedback-Based Protocol(DBP) also introduced in [1] instead of using a fixed leader. The detailed leader election process for a fixed leader is given in [1]. Each protocol is briefly described in the followings.

A. Delayed Feedback-based Protocol

This DBP is devised to avoid the CTS collision by utilizing a random timer. Since DBP entirely relies on the random timer, there is no guarantee that the successful RTS-CTS handshaking is performed within several timer periods.

[A] Base ⇒ Receivers.

1. Send a multicast-RTS.
2. Start a timer (timeout Period T), expecting to hear a CTS before the timer expires.

[B] Receiver ⇒ Base.

1. On hearing RTS, start the timer with an initial value chosen randomly from 1, 2, ..., L.
2. Decrement the timer by 1 in each slot.
3. If a CTS is heard before the timer expires, freeze the timer (CTS suppression).
If no CTS is heard before the timer expires, send CTS.

[C] Base ⇒ Receivers.

1. If no CTS is heard within T, back off and go to Step A.

If a CTS is heard within T (at a random time), start the data transmission.
After completing the transmission, prepare to transmit the next packet and go to Step A (no waiting for feedback).

The next Step is executed only when multicast transmission occurs in step C.

[D] Receivers ⇒ Base.

1. Leader : If packet received without error, send ACK.
If in error, send NAK.
2. Others : If packet received without error, do nothing.
If in error, send NAK.

B. Leader-based Protocol

In this protocol, one of the receivers in the multicast group has been selected to be a fixed leader in order to send CTS and ACK in response to RTS and data packets, respectively. When joining a cell, the base station maintains a table corresponding to each group and chooses a leader. In leader-based protocol in [1], the receiver which first joined the group can be a leader. When this leader leaves the cell or the group, the base station should select another leader explicitly. This sort of leadership maintenance can invoke high overhead. The following procedure is applied after a leader has been elected.

[A] Base ⇒ Receivers (slot 1).

1. Send a multicast-RTS.

[B] Receiver ⇒ Base (slot 2).

1. Leader : If ready to receive data, send CTS.
If not ready to receive data (e.g., due to insufficient buffers), do nothing.
2. Others : If ready to receive data, do nothing.
If not ready to receive data, send NCTS (Not Clear-To-Send).

[C] Base ⇒ Receivers (Slot 3).

1. If a CTS was heard in slot 2, start multicast transmission.
If no CTS was heard in slot 2, back off and go to Step A.

The next Step is executed only when multicast transmission occurs in step C.

[D] Receivers ⇒ Base (Slot (l+3)).

1. Leader : If packet received without error, send ACK.
If in error, send NAK.
2. Others : If packet received without error, do nothing.
If in error, send NAK.

Both ACKs and NAKs from receivers are used as the feedback to the base station. Also, the collisions associated with one or more NAKs are used to ensure that the base station does not obtain a positive feedback if one or more group members received an errorneous transmission.

In [1], the loss of control messages like RTS and CTS is assumed to be ignored. This assumption cannot be in practice. Moreover, LBP doesn't have any mechanism to process with when several consecutive RTS messages cannot reach the leader safely. Although it is likely to be addressed in case that the base station forcely removes the leadership to another member in the cell, LBP doesn't have any efficient mechanism.

C. Random Leader-Based Protocol

Also, both ACK and NAK are used to support the reliable transmission in this protocol. When the base station sends the first multicast-RTS packet, a CTS packet from any of the group members is transmitted when each of their timer value randomly chosen expires. The timer for the receiver whose CTS packet is sent safely to the base station is then set to 0 instead of a random timer value for next rounds. It means that the receiver then becomes a random leader. The detailed description is as follows.

[A] Base ⇒ Receivers.

1. Send a multicast-RTS.
2. Start a timer (timeout Period T), expecting to hear a CTS before the timer expires.

[B] Receiver ⇒ Base.

1. On hearing RTS, start the timer with an initial value chosen randomly from 1, 2, ..., L.
2. Decrement the timer by 1 in each slot.
3. If a CTS is heard before the timer expires, freeze the timer (CTS suppression).
If no CTS is heard before the timer expires, send CTS.
4. If a CTS is sent, set the timer to 0 when the next packet comes.

[C] Base ⇒ Receivers.

1. If no CTS is heard within T, back off and go to Step A.
If a CTS is heard within T (at a random time), start the data transmission.
After completing the transmission, prepare to transmit the next packet and go to Step A (no waiting for feedback).

The next Step is executed only when multicast transmission occurs in step C.

[D] Receivers ⇒ Base.

1. Leader : If packet received without error, send ACK.
If in error, send NAK.
2. Others : If packet received without error, do nothing.
If in error, send NAK.

In this protocol compared to the leader-based protocol, the base station does not need to perform a complex mechanism to maintain the leader table or refresh it whenever the leader leaves. However, this protocol faces the problem of multiple leader occurrences described as the following section. We would like to enhance this protocol to solve the problem caused by multiple leaders.

III. Proposed Protocol

A. Protocol Description

In [2], the authors hold that the RLBP outperforms the LBP because LBP has much overhead for maintaining the leadership management in that the leader in RLBP can be dynamically selected(Refer to [2] for comparison of LBP and RLBP in detail).

However, since [1,2] does not consider the loss of control messages like RTS and CTS, the RLBP has the high probability that multiple leaders can occur in the network and results in the degradation of performance when the loss of RTS and CTS messages occurs.

Suppose an RTS message from a base station to receivers could not reach the existing leader, while some of the receivers heard the RTS successfully. The leader does nothing because the leader doesn't know even that this message is an RTS due to the corrupted message. In the case, a CTS message will be able to reach the base station from one of other receivers after its timer expires if at least one of the selected timer values is different among the others. The receiver will also become a leader. This causes the next slot time to be always wasted due to the collision of CTS messages from the multiple leaders.

If one of receivers(for example, R_n) select its timer as a value greater than slot 0 and there is no other receivers which selected the same value as R_n did, the CTS from this receiver can reach the base station and the base station can transmit a data packet. However, it takes much time to perform the successful data transmission until the timer of R_n expires. It is because there is always collision from multiple leaders at the very next slot time after receiving the RTS message. To make matters worse, the receiver also operates as a leader.

In the worst case, it is likely that all receivers may be leaders in the group after repeating this process. If so, we cannot expect any successful data transmission due to consecutive CTS collisions.

To address this kind of problem, we propose an efficient protocol as follows.

[A] Base ⇒ Receivers.

1. Send a multicast-RTS.
2. Start a timer (timeout Period T), expecting to hear a CTS before the timer expires.
3. Start a timer (timeout Period R = a x T), expecting to hear a CTS before the timer expires.

[B] Receivers ⇒ Base.

1. On hearing cl-RTS(clear-leader RTS), all nodes including Leaders operate in the initial state and start the timer with an initial value chosen randomly from 1,2, ..., L. Otherwise, perform the followings.
2. Random Leaders : 1. On hearing RTS, send CTS. 2. If another CTS is sent after sending its own CTS, stops operating as Leader and begins operating as normal node.
3. Others : 1. On hearing RTS, start the timer with an initial value chosen randomly from 1, 2, ..., L. 2. Decrement the timer by 1 in each slot. 3. If a CTS is heard before the timer expires, freeze the timer (CTS suppression). If no CTS is heard before the timer expires, send CTS. 4. If a CTS is sent, set the timer to 0 when the next packet comes.

[C] Base ⇒ Receivers.

1. If no CTS is heard within T, back off and go to Step A.
If no CTS is heard within R, back off and go to Step A with cl-RTS.
If a CTS is heard within T (at a random time), start the data transmission.
After completing the transmission, prepare to transmit the next packet and go to Step A (no waiting for feedback).

The next Step is executed only when multicast transmission occurs in step C.

[D] Receivers ⇒ Base.

1. Leader : If packet received without error, send ACK.
If in error, send NAK.
2. Others : If packet received without error, do nothing.
If in error, send NAK.

B. Illustrative Example

In this part, we would like to illustrate an example of the proposed protocol. We assume that there are five receiver members in a cell as shown in Figure 2. We don't

show other nodes which are not group members in the figure. First, a base station(BS) sends a multicast RTS message when it wants to send a multicast data packet. All nodes from node 1 to node 5 receive the RTS message safely(Figure 2a). All nodes select their random timers for sending a CTS message when the timers expire. As shown in Figure 2b, fortunately, node 2 could send the CTS message to the BS due to the absence of collision of CTS messages from other nodes. In this case, node 2 becomes a leader. Therefore, when the BS sends another RTS message, node 2 sends its CTS message and suppresses other CTS messages generated by other nodes in the next slot right after the leader receives the RTS.

However, in case that an RTS message cannot reach node 2 as shown in Figure 2c, one of other nodes sends its CTS message safely because its random timer expires and furthermore, there is no collision of CTS message in the T slots. It allows node 3 to send the CTS message and become also a leader(Figure 2d). However, node 2 can hear the CTS message generated by the node 3 safely and relinquish its role of a leader in order to reduce the possibility of occurrence of multiple leaders.

Otherwise, as shown in Figure 2e and 2f, the node 3 receives a next RTS and sends the corresponding CTS message safely to the BS.

Suppose that the node 2 could not hear the CTS message safely due to some other conditions(i.e., busy in other computation). In this case, even if there exist multiple leaders in the cell, one of other nodes can become a leader when its random timer expires after wasting the next slot right after node 2 and 3 receive the RTS message due to the collision of CTS messages. In case that this kind of situation continues, all nodes seem to be all leaders. Therefore, the BS sends a cl-RTS message and all nodes depend on their random timers when they receive the cl-RTS and also relinquish their roles of leaders(See Figure 2g, 2h and 2i).

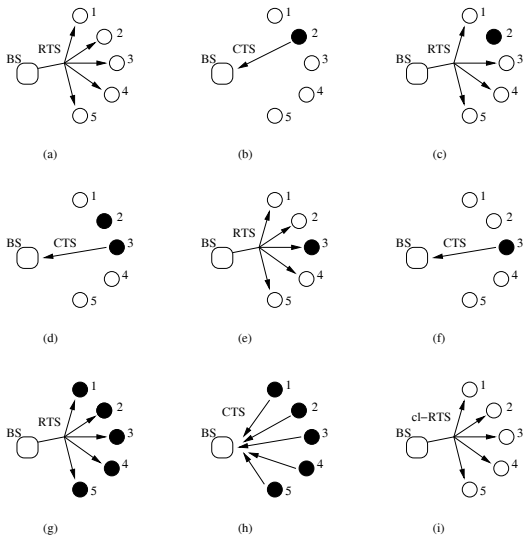

Fig. 2. Example.

IV. Performance Evaluation

A. Simulation Environment

In this work, we compared the RLBP and our proposed protocol when the control messages like RTS and CTS are prone to loss. In other words, while some of receivers can receive the control messages safely, the others of the receiver members cannot. It allows multiple leaders to occur frequently, resulting in the preference of our proposed protocol over the RLBP.

We developed an event-driven simulator where a cell is assumed. In the cell, all nodes including the base station are within the radio transmission ranges of one another, which causes no hidden terminals.

N nodes are spreaded in the cell. The base station is located in the center of the cell. All multicast communication should be conducted from the base station to the receiver members.

None of interframe space like DIFS and SIFS is implemented in the simulator, for simplicity. Each time is slotted, when a slot time is assumed to be 1 simulation tick. While the control messages consume 1 slot time, the size of transmitted packet is fixed which spends 6 slot times by including the ACK transmission. In order to determine R value, a is set to 5.

For comparing and analysing LBP and RLBP, refer to [2] which shows that RLBP outperforms LBP. Therefore, in this work, we simulate and compare RLBP and our protocol in terms of the possibility of the occurrences of multiple leaders.

B. Observed Results

First, we investigate the average time until all nodes become the leaders and the multicasting activity cannot progress any more in random leader-based protocol. As mentioned before, the RLBP doesn't address the occurrences of multiple leaders. It is because RLBP ignores the loss of RTS message. Although LBP also ignores the loss of RTS message, LBP doesn't have multiple leaders in a cell. However, there exists much overhead for maintaining the leader which the RLBP was proposed to solve.

By considering the possibility of loss of RTS message, as the loss of the RTS message increases at the leader, the

period during which at least one of receiver members is still not a leader and is able to send its CTS message to a base station safely decreases. This period lasts much longer when the number of receiver members is high(See Figure 3).

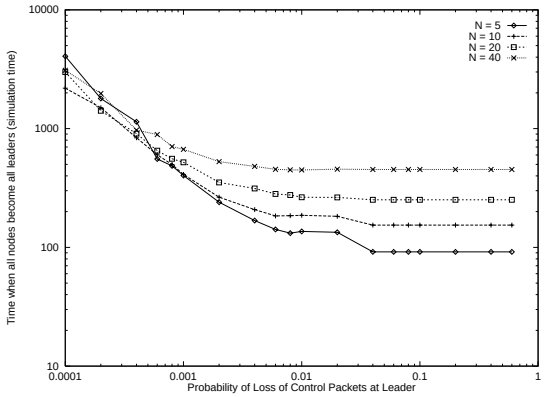

Fig. 3. Average Time Until All Nodes become All Leaders : RLBP, T = 5, L = 30.

As another simulation, we measured the average access time until a base station has the chance to send its data packet upon receiving the corresponding CTS message after transmitting its RTS message. As for this average access period, we compare our protocol with RLBP. In case that some multiple leaders exist in RLBP, the slot which the leaders try to access for transmitting their CTS messages is not useful due to their collision. However, RLBP can expect some success within T slots for allowing the base station to transmit its data packet. The node A which transmitted the successful CTS becomes a leader. Therefore, when the base station sends another RTS message for its next data transmission, the same problem should be addressed.

Our protocol allows the node A to become a leader and the other leaders to relinquish their roles on hearing the CTS message from the node A safely. It result in the node A responding to the next RTS message generated by the base station when the base station wants to transmit other packets. It can reduce the access time before transmitting data packets(See Figure 4).

Also, we observed how T has impact on the performance, especially, the average access period in our proposed protocol. As shown in Figure 5, we acquired the result that for all Ts, as the probability of loss of RTS message increases, the average access period also increases as investigated in the simulation above.

In addition, intuitively, we can consider that a larger T gives some nodes the higher chances to send their CTS messages safely within T even if multiple leaders waste the first slot in the T due to their collision. However, when every trial fails in sending the CTS message, the larger T wastes more slot times until the base station retransmits its RTS

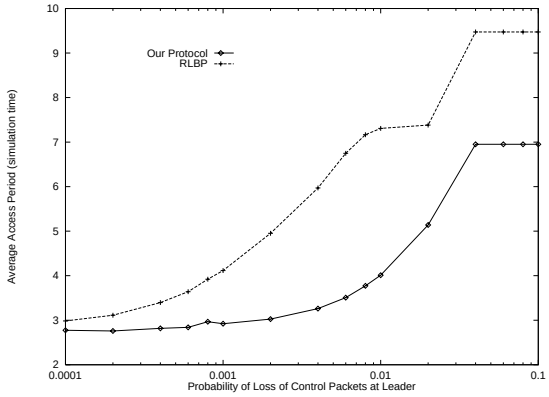

Fig. 4. Comparison of Average Access Period : Our Protocol and RLBP, N= 10, T = 5, L = 30.

message. In this simulation(See Figure 5), the reason why the higher T has the higher access period is that the more collisions occurred within the T slots instead of successful CTS transmission.

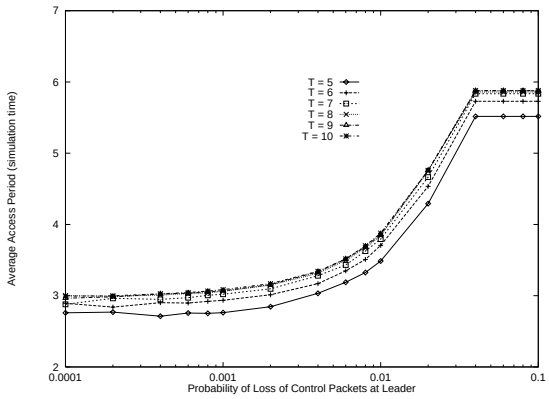

Fig. 5. Comparison of Average Access Period According to T : Our Protocol, N= 20, L = 30.

Finally, we also measured the number of data packets transmitted from a base station during a given simulation time according to various L values. We can easily derive that as the control packet is more lossy, the number of packets transmitted decreases irrespectively of L values. However, from the simulation result(See Figure 6), we could observe that a larger L could send more data packets than a smaller L could. We can conjecture that when L is large, the number of nodes with the same slot time for their CTS messages within the T slots decreases, resulting in increasing the probability of successful CTS transmission in the T slots. On the other hands, we should not ignore some higher probability that for a larger L, all nodes select larger values than T as their random timers because of $T < L$. It results in the degradation of performance. However, it seemed that the condition occurred in the simulation when L = 100.

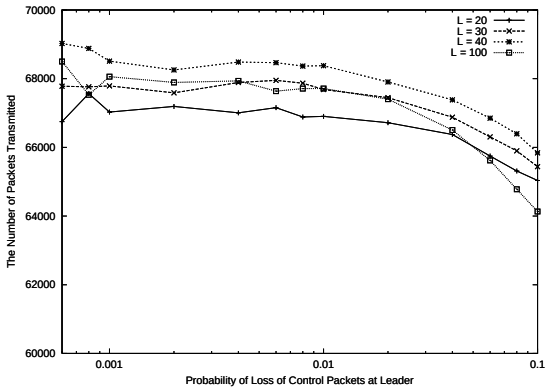

Fig. 6. The Number of Packets Transmitted According to L : Our Protocol, N= 20, T = 5.

V. CONCLUSIONS

Many research groups have paid attention to the importance of multicasting protocol in the wireless networks in terms of the reduction of network resources. Specially, a reliable multicasting protocol shoud be devised to support the efficiency of reliable group communications.

In a wireless LAN, the handshaking of RTS-CTS-DATA-ACK achieves the reliability of a unicast communication. However, this kind of handshaking is not suitable for a multicasting support due to the problem of CTS and ACK explosion. Although two protocols like leader-based(LBP) and random leader-based protocol(RLBP) are addressing the problem, they still have the overhead of leader maintenance and in RLBP, particularly, it has the possibility of multiple leaders in a cell. To overcome these problems, we proposed a more efficient reliable multicasting protocol in this work. It borrows the concept of the RLBP and however, avoids the communication breakage caused by multiple leasers.

REFERENCES

[1] J.Kuri and S.K.Kasera, "Reliable Multicast in Multi-Access Wireless LANs," Wireless Networks 7, pp. 359-369, 2001.

[2] H.C.Chao, S.W.Chang, and J.L.Chen, "Throughput Improvements Using the Random Leader Technique for the Reliable Multicast Wireless LANs," ICN 2001, LNCS 2093, pp. 708-719, 2001.

[3] B.P. Crow et al., "IEEE 802.11 Wireless Local Area Networks," IEEE Commun. Mag., Vol. 35, no. 9, Sept., 1997.