An Idea of Linux Implementation of Fixed Backoff-time Switching Method for Wireless Mesh Networks

Sritrusta Sukaridhoto[1] Nobuo Funabiki[1] Toru Nakanishi[1] Kan Watanabe[1] Shigeto Tajima[2]

Graduate School of Natural Science and Technology, Okayama University[1]
Graduate School of Engineering Science, Osaka University[2]

Abstract

As a flexible and cost-efficient scalable Internet access network, we have studied architectures, protocols, and design optimizations of the *Wireless Internet-access Mesh NETwork (WIMNET)*. WIMNET is composed of wireless connected access points. Previously, we proposed the *Fixed Backoff-time Switching (FBS) method* for the CSMA/CA protocol to improve the real-time traffic performance in WIMNET by giving the necessary activation chances to each link, and verified the effectiveness using the *QualNet* simulator. In this paper, we present an idea of the FBS method implementation at Linux kernel after quickly reviewing it.

1 Overview of FBS Method

The FBS[1] method selects either of the two backoff-times, namely, the shorter *active backoff-time* and the longer *passive backoff-time*, for each link transmission by comparing the *target link activation rate* and the *actual link activation rate*, so that the link can be activated to handle the required traffic. Any backoff-time is assigned a different fixed value so that no pair of the conflicting links may be activated simultaneously. Besides, the backoff-time for a link with larger traffic is assigned a smaller value than that for a link with smaller one, so that congested links can be activated preferentially.

During communications, every time a node holding packets detects that the channel becomes free, it updates both the target activation rate and the actual activation rate. If the actual one is smaller than the target one, it selects the active backoff-time to let the link be activated, because the current activation rate of the link is not sufficient to handle its traffic. On the other hand, if it is larger, it selects the passive backoff-time to let other links with active backoff-times be activated with higher priorities. A link with the passive backoff-time can be activated only if any conflicting link with the active backoff-time does not hold packets. The following sections describe how to calculate the parameters in the FBS method.

2 Target Link Activation Rate

For a wireless link l_{ij} transmitting packets from AP_i to AP_j for $i = 1, \cdots, N$ and $j = 1, \cdots, N$ in WIMNET with N APs, the target link activation rate rt_{ij} can be calculated by:

$$rt_{ij} = \frac{tn_{ij}}{an_{ij}} \tag{1}$$

where tn_{ij} represents the target number of activating link l_{ij} per second, and an_{ij} does the average number of link activations per second. tn_{ij} can be given from the requested bit rate by:

$$tn_{ij} = \frac{rb_{ij}}{fb_{ij}} \times (1 + fe_{ij}) \tag{2}$$

where rb_{ij} represents the number of bits per second that link l_{ij} needs to be transmitted, fb_{ij} does the average number of bits in one transmitted frame, and fe_{ij} does the rate of causing the frame transmission error. an_{ij} can be given by:

$$an_{ij} = \frac{1}{ft_{ij}} \tag{3}$$

where ft_{ij} represents the average duration time of one frame transmission.

Among the parameters for the target link activation rate, rb_{ij} should be calculated by taking the summation of the bit rates requested by the applications using link l_{ij} in the routing path of WIMNET. The others, fb_{ij}, fe_{ij}, and ft_{ij}, should be updated during communications by the following equations:

$$fb_{ij} = \frac{sb_{ij}}{sf_{ij}} \tag{4}$$

$$fe_{ij} = \frac{ff_{ij}}{sf_{ij} + ff_{ij}} \tag{5}$$

$$ft_{ij} = \frac{t}{sf_{ij} + ff_{ij} + of_{ij}} \tag{6}$$

where sb_{ij}, sf_{ij}, ff_{ij}, and of_{ij} represent the total number of successfully transmitted bits by link l_{ij}, the total number of successfully transmitted frames, the total number of failed frames, and the total number of transmitted frames of the interfered links with link l_{ij}, when t seconds have passed since the communication started in WIMNET, respectively.

During communications, the values of rb_{ij}, fb_{ij}, fe_{ij}, and ft_{ij} should be automatically updated by using the obtained values of sb_{ij}, sf_{ij}, ff_{ij}, and of_{ij}. For this purpose, we uses $minstrel$[3] data rate structures. The *success* variable represent sb_{ij} and sf_{ij} when AP receives an ACK message. Besides, *retry_count* variable value updates ff_{ij} when AP retransmits a failed frame. To obtain of_{ij}, we uses the summation of *retry_count_cts* and *retry_count_rtscts*. And we uses function *get_sta_rates* from *userspace* to update rb_{ij} value.

3 Actual Link Activation Rate

The *actual link activation rate* ra_{ij} for link l_{ij} is obtained by dividing the number of successfully transmitted frames with the number of possibly activating chances for the link:

$$ra_{ij} = \frac{sf_{ij}}{ac_{ij}} \tag{7}$$

where ac_{ij} represents the number of possibly activating chances of link l_{ij}.

ac_{ij} is counted every time AP_i detects that the channel becomes free. We uses *attempts* variable to get the value of ac_{ij} from *minstrel* data rate structures.

4 Active/Passive Backoff-time

The *active backoff-time* ta^m_{ij} and the *passive backoff-time* tp^m_{ij} for link l_{ij} are calculated by the following procedure.

1. Calculate the number of bits to be transmitted per second rb_{ij} for link l_{ij} by taking the summation of the bit rates for all the communication requests by the hosts using l_{ij}:

$$rb_{ij} = \sum_{k \in H_{ij}} hr_k \tag{8}$$

where H_{ij} represents the set of the host indices using link l_{ij} in the routing path, and hr_k does the requested bit rate (bps) of host k.

2. Sort every link in descending order of rb_{ij}, where the tiebreak is resolved by the number of hosts using this link for the routing path.
3. Set this sorted order to the link priority p_{ij} for l_{ij}.
4. Calculate the active/passive backoff-times for l_{ij}:

$$\begin{aligned} tamin^m_{ij} &= CW_{\min} \cdot \left(2^{m-1} + 2^{m-2} \cdot \tfrac{p_{ij}-1}{P}\right), \\ tamax^m_{ij} &= CW_{\min} \cdot \left(2^{m-1} + 2^{m-2} \cdot \tfrac{p_{ij}}{P}\right), \\ ta^m_{ij} &= rand\left[tamin^m_{ij}, tamax^m_{ij}\right], \end{aligned} \tag{9}$$

where $tamin^m_{ij}$ and $tamax^m_{ij}$ represent the minimum and maximum values for the active backoff-time for l_{ij} when the retry counter is m, $CW_{\min}$ does the initial CW size, and P does the largest priority among the links.

$$\begin{aligned} tpmin^m_{ij} &= CW_{\min} \cdot \left(2^{m-1} + 2^{m-2} \cdot \tfrac{P+p_{ij}-1}{P}\right), \\ tpmax^m_{ij} &= CW_{\min} \cdot \left(2^{m-1} + 2^{m-2} \cdot \tfrac{P+p_{ij}}{P}\right), \\ tp^m_{ij} &= rand\left[tpmin^m_{ij}, tpmax^m_{ij}\right]. \end{aligned} \tag{10}$$

where $tpmin^m_{ij}$ and $tpmax^m_{ij}$ represent the minimum and maximum values for the passive backoff-time for l_{ij} when the retry counter is m.

5 Linux Implementation of FBS Method

In this section, we present an idea of the FBS method implementation at Linux kernel [2]. The FBS method uses a fixed value for CW_{min}. Thus, we need to disable the random backoff-time in Linux kernel, and assign the active/passive backoff-time. We may implement it as *daemon* in the application layer. It obtains the required parameters for the FBS method as *minstrel*. Then, after calculating the FBS parameters, it gives the backoff-time values as $CWmin$ in *userspace*, shown in Figure 1.

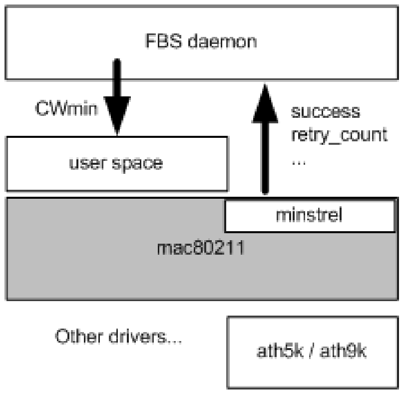

Fig. 1 FBS Daemon.

Acknowledgments

This work is partially supported by KAKENHI (22500059).

References

[1] S. Sukaridhoto, N. Funabiki, T. Nakanishi, and K. Watanabe, "A proposal of CSMA fixed backoff-time switching protocol and its Implementation on QualNet simulator for wireless mesh networks", Proc. WAINA, March 2012.

[2] M. Vipin and S. Srikanth, "Analysis of open source drivers for IEEE 802.11 WLANs", Proc. ICWCSC, 2010.

[3] Minstrel - Linux Wireless, http://linuxwireless.org/en/developers/Documentation/mac80211/RateControl/minstrel/.