**Weekly Report**

We did a series of experiments on the last two weeks, focusing on analyzing some built-in parameters collected by Windows Native Wi-Fi API. Experiment setup: 802.11g Cisco Linksys AP /microwave oven / Bluetooth earphone. We did the experiments on Kyung’s office (7th floor of CEPSR). In the experiment, we use one 802.11g AP (working on channel 9) to connect to internet and download a large file from the internet. We fix the 3-meter distance between AP and labtop; change the location of AP to vary its distance from injected interference sources. The measurements are obtained with only one labtop connecting to the AP. (We cannot find a place to eliminate the influence of APs working on the same channel and neighboring channels.) The query interval is 200ms and the measurement lasts for 1 minute.

We hope to find some meaningful patterns of the *percentage of retransmitted frames*, *number of successfully transmitted frames*, *ACK failure counts*, *signal strength*. We make some preliminary conclusions of our observations.

Fig-1,-2,-3 show the influence caused by Microwave Oven. It is clearly seen that Microwave Interference results in more retransmitted frames.

When AP is placed right above the Microwave oven, as shown in Fig-1, mean=0.314, variance=0.106, compared to the cases without interference where mean=0.205, variance=0.013.

When AP is 2 meters away from the Microwave oven, as shown in Fig-2, mean=0.362, variance=0.037

When AP is 5 meters away from the Microwave oven (partly blocked by the wall), as shown in Fig-3, mean=0.237, variance=0.008.

To conclude, the interference is damped down as the distance increases. If the variance is over 5 times higher than the normal and the mean goes higher simultaneously, the interference source should be located nearby. The same scenario happens as the Microwave Oven is placed 2 meters away from the AP with no blockings. As the AP becomes farer from the Microwave, the percentage of retransmitted frames backs to the normal case.

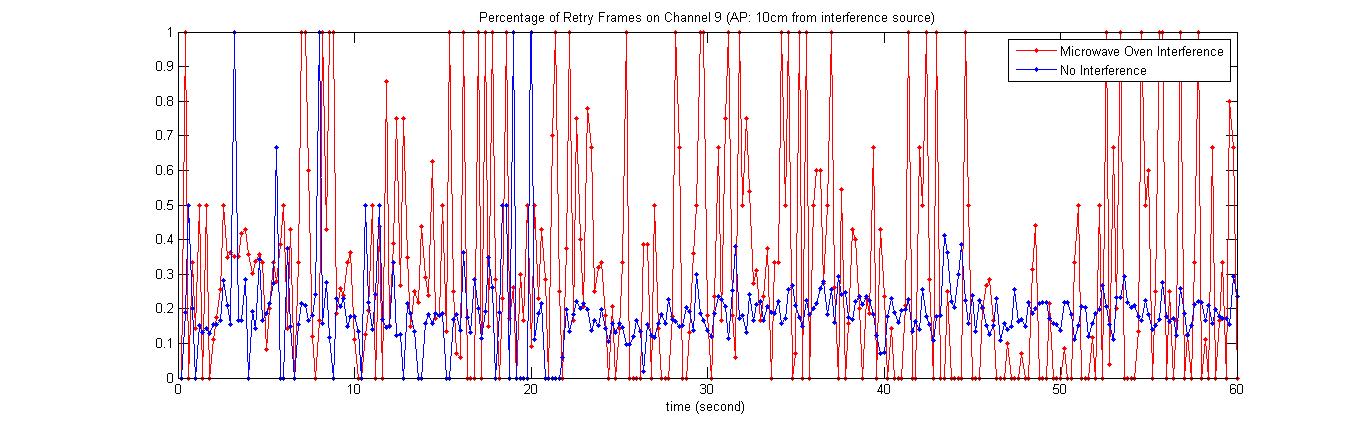


Fig-1

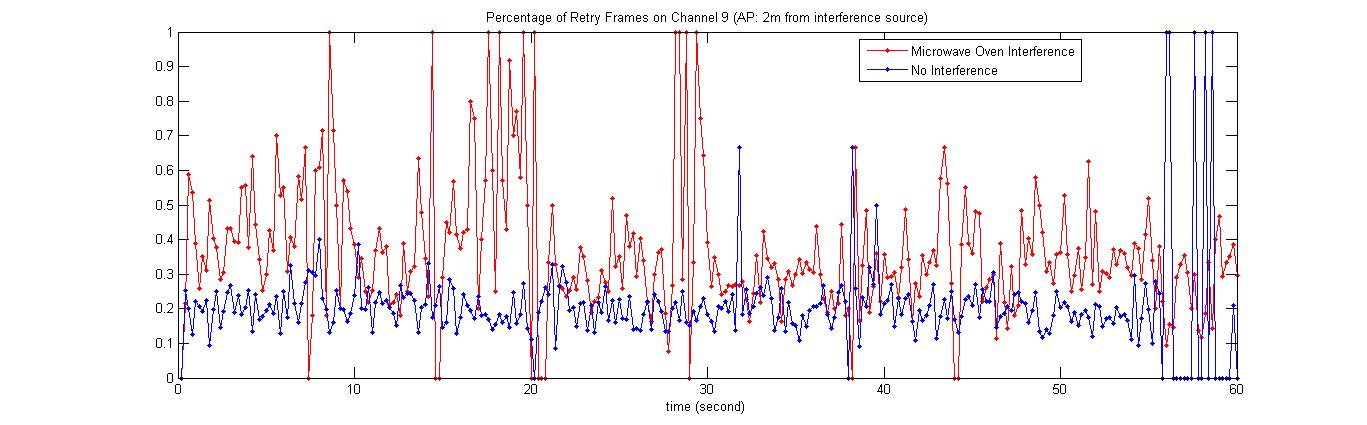


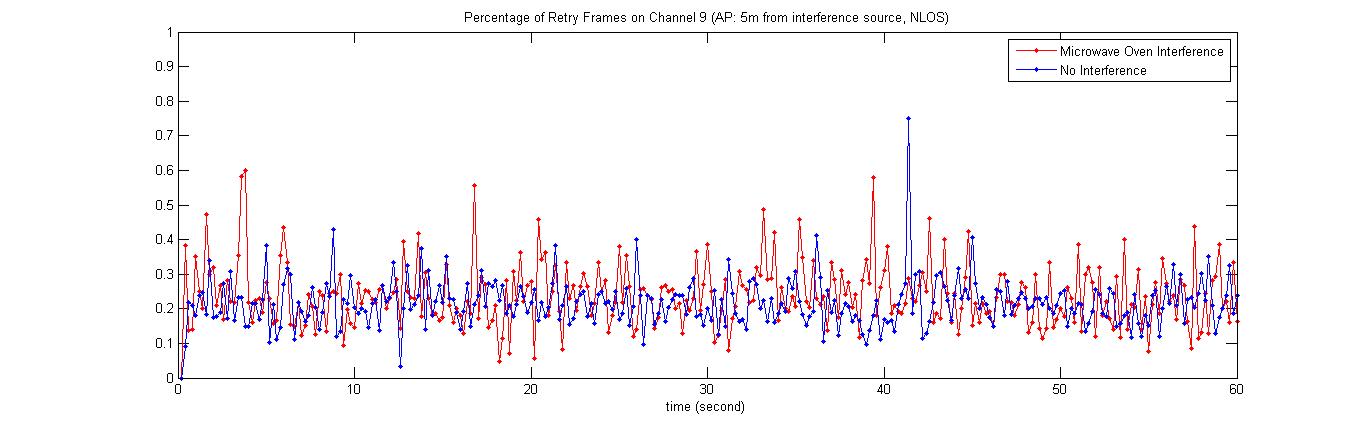
Fig-2

Fig-3

We also try to turn on the Microwave Oven in the middle way of our experiment. The measurement is shown in Fig-4. Again we observe large fluctuation after turning on the Microwave oven. The short period of extreme fluctuation is sort of misleading, because the total transmitted frames (denominator on the calculation) becomes extremely fewer, say 0, 1, 2, when turning on the Microwave oven.

According to what we learned from other paper, Microwave Oven works like ON/OFF every 5 to 6 ms, such pattern can be easily found through spectrum analyzer. We are wondering if similar characteristics can be captured on time domain, as 6 ms is quite short interval.

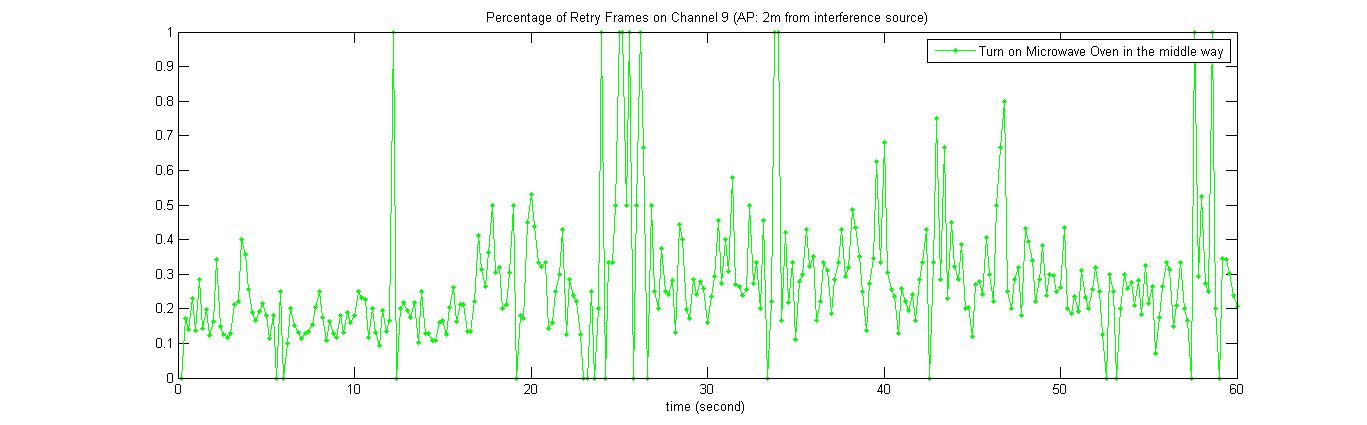


Fig-4

We also did some experiments on the injected Bluetooth interference, putting the Bluetooth earphone near AP/ Labtop and using it for music playing. Unfortunately, we cannot observe an obvious retry behavior to distinguish Bluetooth interference. (The extreme fluctuation shown in Fig-5 can be explained as a sharp drop of the total transmitted frames, therefore the percentage changes extremely.)

The following reasons might explain our results.

1. The transmitted power of Bluetooth earphone is lower.
2. Bluetooth uses DSSS technique that hops frequently and randomly in the 2.4GHz.

(It is found in the other paper that Bluetooth devices do not cause severe throughput degradation.)

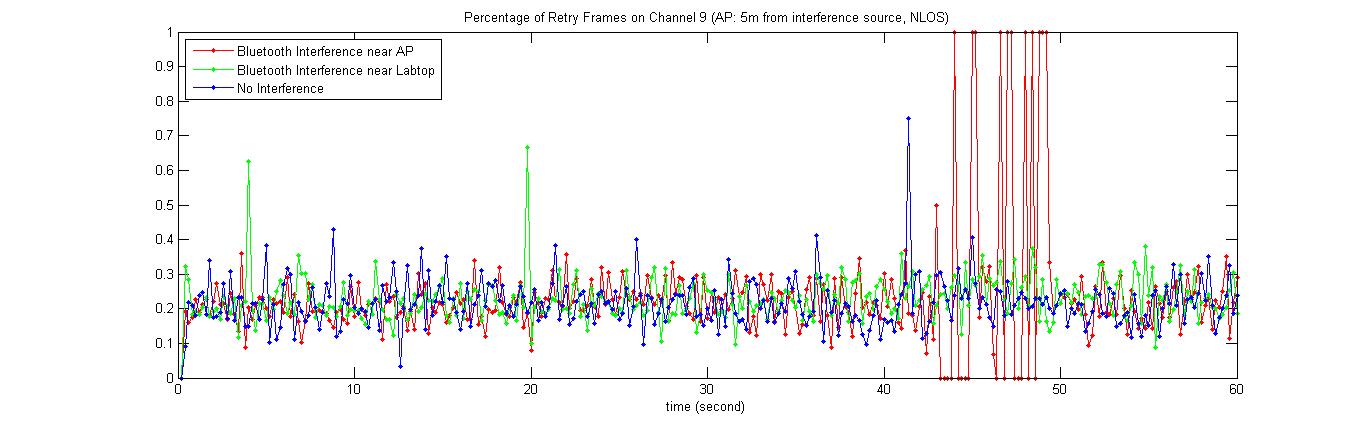


Fig-5

Fig-6, -7, -8 show the number of successfully transmitted frames and the corresponding ACK failure counts with/without Microwave oven interference. The blue line shows normal cases; the red line shows cases where Microwave oven interference is injected.

As we know, in the legacy 802.11 a/b/g systems, an ACK frame is sent from the receiving station to the transmitting station to confirm the reception of each frame. If the transmitter does not receive an ACK, it retransmits the frame until an ACK is received. The Native Wifi API returns the number of times that the 802.11 station expected and did not receive an ACK frame. It is intuitively that the ACK failure counts should behave differently under different interference.

Carefully observing the results we found that, the ACK failure times without injected interference are mostly less than the number of successfully transmitted frames; while incorporating the Microwave Interference, the ACK failure times outweighs the number of successfully transmitted frames on some sample intervals.

In particular, we found less than 10% sample points having larger number of ACK failures than that of successfully transmitted frames. However, when AP is close to the Microwave oven, 87% sample points have more ACK failures than the successfully transmitted. It lowers to 54% and 24% as we move the AP 2 meters and 5 meters (partly blocked by the wall) away from the Microwave oven. As for Bluetooth interference, again we cannot find the difference.

We compare the number of successfully transmitted frames with the number of ACK failures in order to find the pattern of ACK failures changing with the interference, but we are not sure whether this makes sense or not. There could be other ways to evaluate the ACK behaviors.

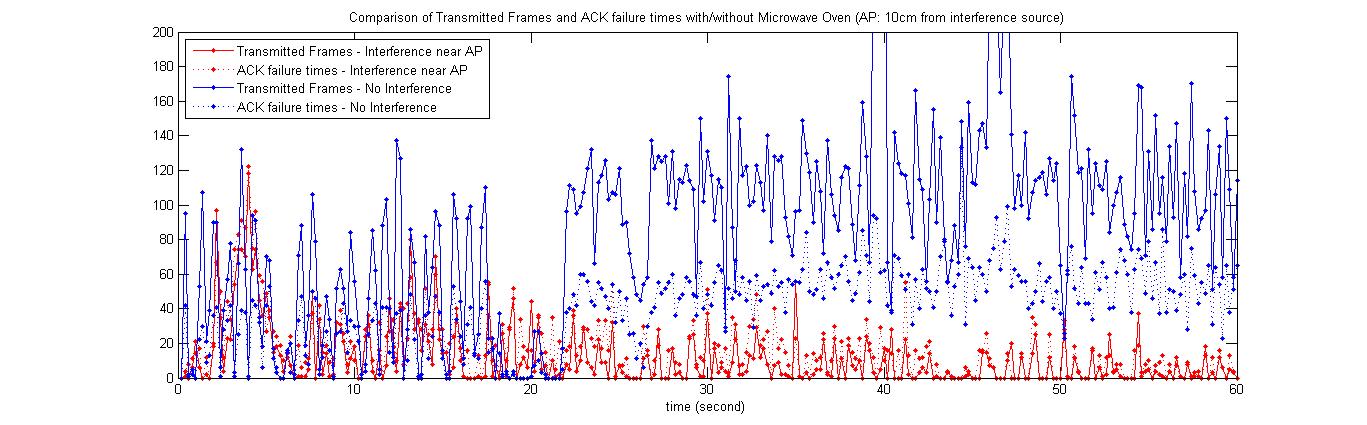


Fig-6

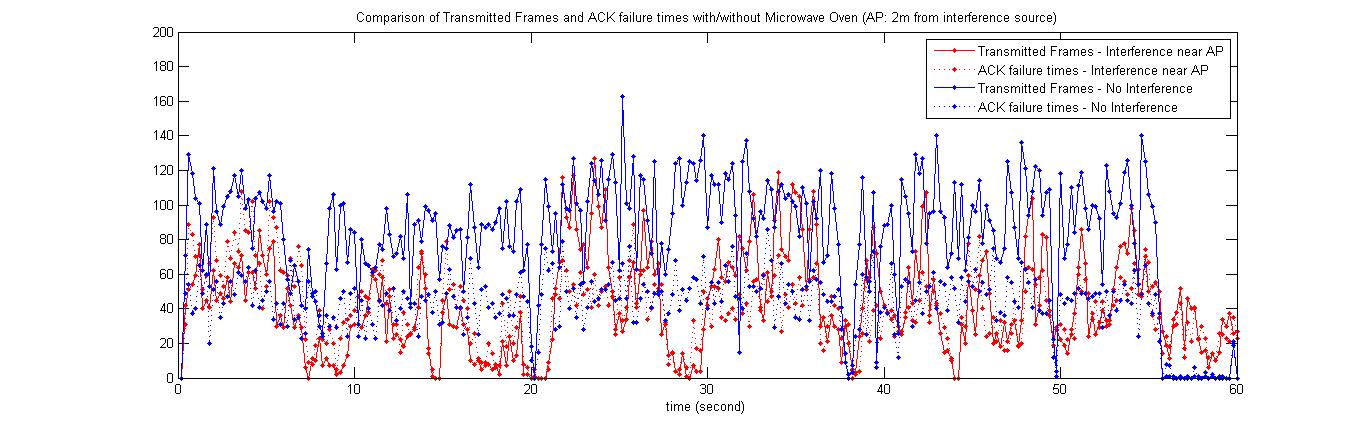


Fig-7

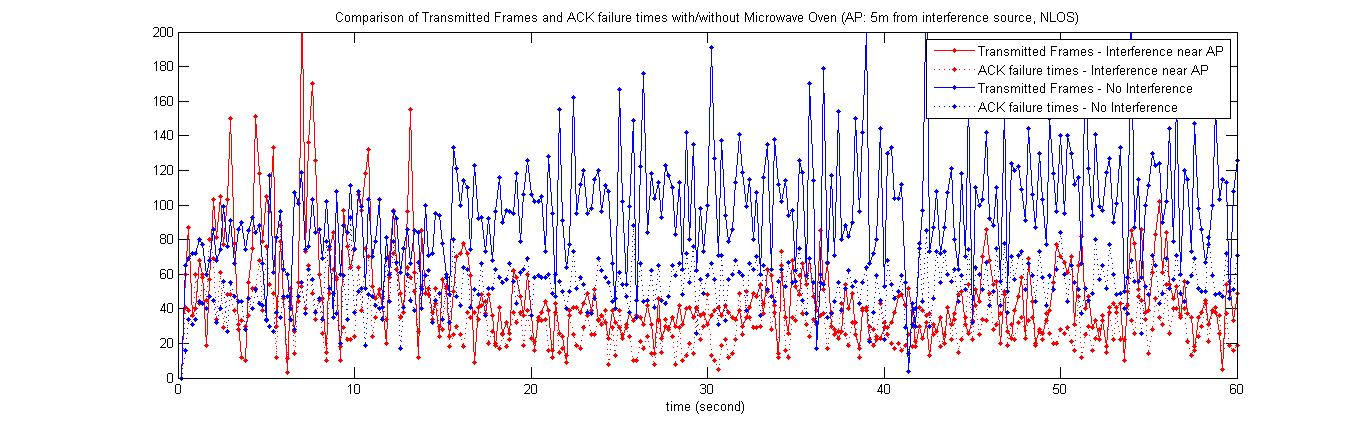


Fig-8

In Fig-9, we see a general better performance of Bluetooth interference than Microwave oven interference. In fact, up till now, no parameter indicates the performance degradation of Bluetooth interference; it seems Bluetooth interference is not prominent.

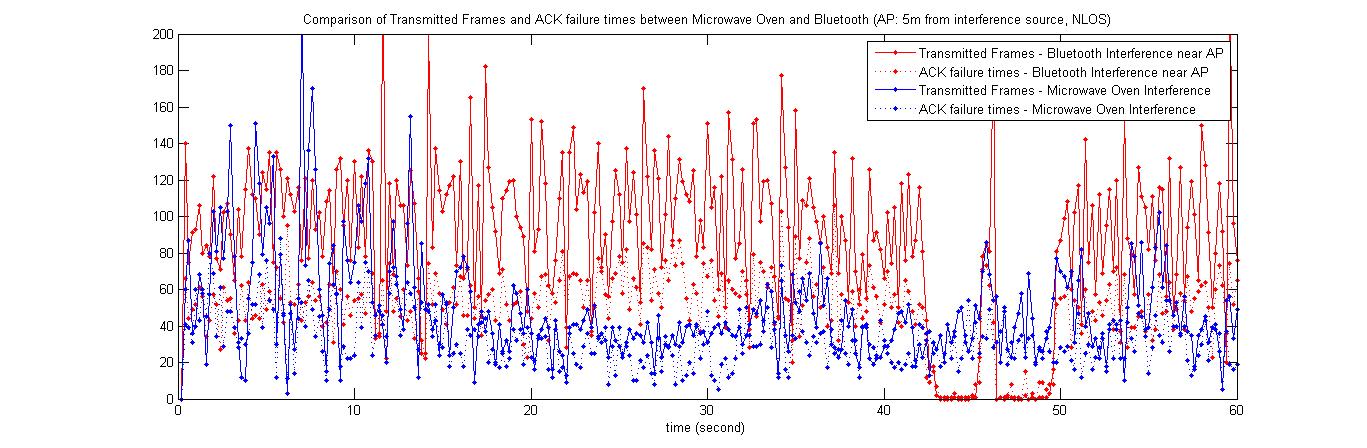


Fig-9

We also gather the statistics of what percentage of ACK failures account for the total attempted transmissions, as shown in Fig-10. The calculation is based on

(ullACKFailureCount) / (ullTransmittedFragmentCount+ ullACKFailureCount)

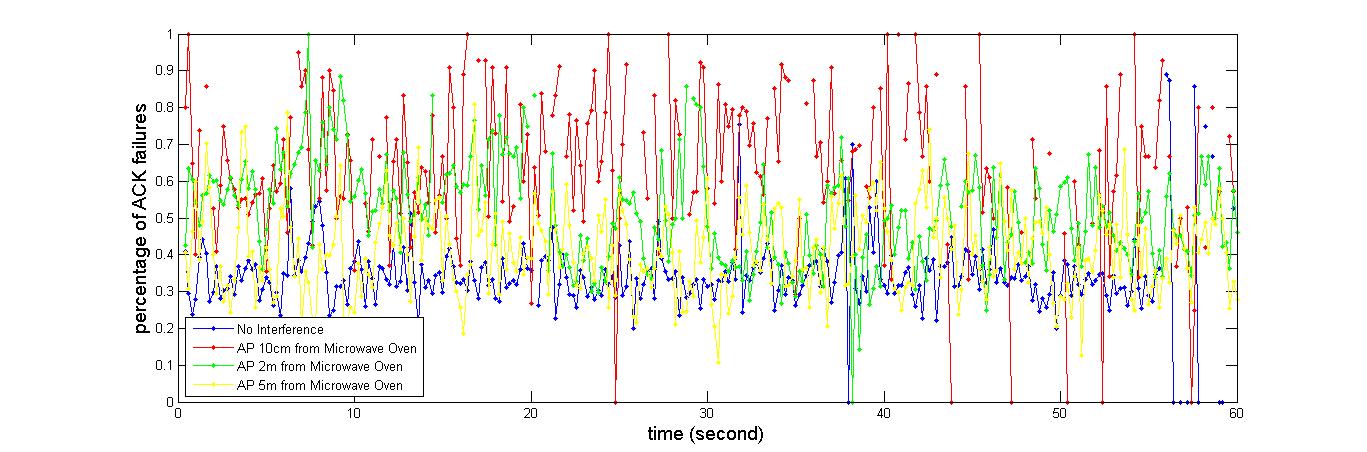


Fig-10

We do not show signal quality related figures here as the signal strength shows no big difference with or without Microwave interference. But the fact is the larger the interference, the lower the SNR. The maximum data rate is partially determined by SNR. So we hope to find some *data rate* related patterns. For now, the Native WiFi API do not support data rate, so we plan to obtain and analyze it in Linux and repeat our experiments.

Questions:

Since the Bluetooth interference is hard to be identified, maybe we can do more experiments to identify the collisions introduced by channel contention. Collisions should trigger a different retry behavior than loss due to noise. What is the expected difference?

For now, we cannot find an ideal place to do our experiments. Although we can switch the working channel of the AP, the neighboring channel interference still exists. There are so many APs nearby and they work in various channels. What should we do about it?