Empirical Evaluation and Analysis of Application-Layer Delay Reduction Methods over Wireless Access Networks

Yoshiaki Nishikawa, Takashi Oshiba, Dai Kanetomo, and Kazauki Nakajima

NEC Corporation. 1753 Shimonumabe, Nakahara-ku,
Kawasaki, Kanagawa, 211-8666 Japan
y-nishikawa@@ah.jp.nec.com,oshiba@cp.jp.nec.com,
d-kanetomo@ce.jp.nec.com,nakajima@ah.jp.nec.com

Abstract. In this paper, we discuss the experimental results of application-layer delay reduction methods which can reduce the delay during TCP applications. Along with the popularization of smart phones, it is increasing in use of WLAN (Wireless Local Area Network) access such as WLAN hot spot or WLAN tethering. This increasing makes a WLAN channel be shared by multiple data flows and results in the degradation of the communication performance. Especially, the delay of the real time application which communicates over TCP can be made longer. In order to reduce the delay, we develop three methods: reducing the number of data to send, triggering TCP's fast retransmission earlier and to returning TCP's acknowledgment immediately. In our experiments, we measure the delay during the real time TCP applications connected with each other over WLAN and commercial fixed network and analyze the effectiveness of our methods in the case that the performance of WLAN is degraded by the file transfer flow. Results indicate that our methods can reduce the 95 percentile of the delay as much as 84%.

Keywords: TCP, Delay Reduction, Application-Layer Method

1 Introduction

Along with the popularization of smart phones, it is increasing in the use of WLAN access. Internet access over WLAN is offered to in various places such as in the home, office, hotels and so on. Mobile devices such as smart phones and tablet devices on which only wireless LAN is available have been popular. WLAN is facing increasing accesses by mobile devices in addition to PCs. It has also became popular to access the Internet through mobile PCs, tablet devices or gaming devices by using a mobile router or the WLAN tethering function on a smart phone. A mobile router is the router that is connected to the Internet via mobile wireless communication technology such as 3G and LTE and has the function as a WLAN access point. The WLAN tethering is to use a smart phone as a mobile router. People are establishing their each WLAN on their demands and are connecting their mobile devices to the Internet.

There are several problems which degrade the performance of WLAN and real time application. Competing and interfering problems occur, because WLANs which do not cooperate with each other are established side by side. The competing problem is the problem that the performance is degraded by the impact from other WLANs which is sharing the same communication channel. The interfering problem is the problem that the performance is degraded by the impact from other WLANs which is overlapping in the spectrum of the communication channel. There is the problem about the growing traffic. More traffic is offloaded from mobile wireless communication system to WLAN. These problems have a significant impact on the real time application which communicates frequently over TCP because of its retransmission and reordering.

In this paper, we propose the delay reduction methods on the real time application and perform the experiment to evaluate these methods in the case that the performance of WLAN is degraded by the file transfer flow. We propose three methods: to reduce the number of data to send, to trigger the TCP fast retransmission earlier and to return an ACK immediately. We try to reduce the delay to less than 100ms which is required as the delay of a real time application[8]. Results show that our methods can reduce the 95th percentile of the delay as much as 84% and achieve the requirement.

2 Related Work

In this section, we introduce other's work related in this paper. There are some work to manage the competing and interfering problem. The channel selecting algorithm which scan channels used by other access points and select own channel automatically is proposed in [3]. In [2], the approach to optimize access point placement and channel assignment in WLANs is proposed. There is the work which shows the results about the delay of the real time TCP application from the experiment[4]. According to that paper, the method which splits data and uses multiple TCP connections can reduce the delay in the managed environment. In this paper, we evaluate our three methods in the case that the performance of WLAN is degraded by the file transfer flow.

3 Application

In this section, we explain the real time application we use. We use the Android device as a platform on which the application and our methods run. This is because Android devices are widely used to access to the Internet. As the real time application, we select the application which sends and receives the small size data that indicates user's operation on the touch panel. A remote desktop application is an example of the application we select. We are interested in the End-to-End one way delay which is the time that it takes data to travel across the network from source to destination. This is because the responsiveness of the real time application depends on how short the End-to-End delay is. For example in remote desktop application, a local device sends data indicating

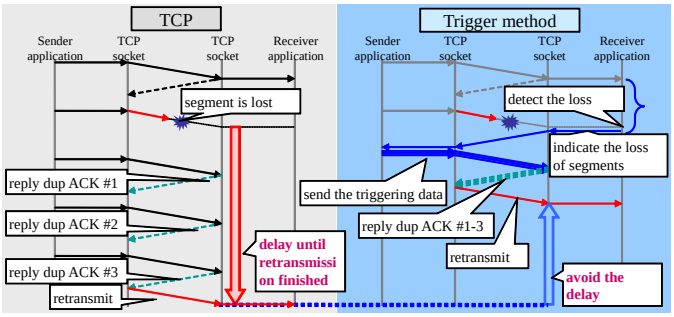

Fig. 1. The sequence of TCP' fast retransmission on the left of the figure and the sequence when using Trigger method on the right.

user's operation to a remote device and receives the result image from remote device. The End-to-End delay of operation data become shorter, users can get the result image more earlier. So it is important to reduce the End-to-End delay of operation data. In [8], 100ms is required as the End-to-End one way delay of real time application. In the following of this paper, we refer to the delay as the End-to-End one way delay.

We suppose that two of TCP's controls make the delay long. One is the reordering control to reconstruct the sending order on the receiving side. TCP performs the retransmission when TCP detects a loss of a segment. Throughout this paper, we refer to the transmission unit of TCP as a segment and to the transmission unit of application as a data. By the reordering control, the segment which is send after the dropped segment is waited at the buffer of the receiving side until the lost segment has been transmitted safely. Then, the delays of data corresponding to the waited segments are extended. More data is delayed by the reordering control in the real time application than in the non real time application which sends data not so frequently. Second is the flow control to avoid the buffer overflow at the receiving side. TCP limits the number of segments which TCP is able to send without receiving ACK. After reaching the limit, TCP delays sending till receiving ACK at the sending side. At the receiving side, TCP also delays replying ACK until the timer have reached a certain timeout value or the receiver has received multiple segments. While TCP delays replying ACK, the number of segments which have been sent already by the sending side is more likely to reach the limit in the real time application than in the no real time application.

4 Methods

In this section, we propose three delay reduction methods. First, the application can reduce the number of data to send and its consumption of bandwidth. We call this data reducing method as Reduction-P method, where P is the factor

representing reduction percentage. To send segments frequently can make the long queuing time in the network and the delay by congestion and reordering increased. This method reduces the times and the total size of sending data to expand the interval between sending events so that the number of segments in the network and the number of congested segments can be reduced.

Second, the application can trigger retransmission faster by sending several data and reduce its delay. We call this data sending method as Trigger method. Figure 1 illustrates the sequence of TCP fast retransmission using TCP on the left of the figure and the sequence using Trigger method on the right. TCP retransmits the dropped segment when multiple (commonly two or three) dup ACKs have been received. The dup ACK is generated at the receiving side and replied to the sending side when the segment is received in wrong order. In Trigger method, the receiver infers the loss of segments from the too long interval of receiving data. Then the receiver sends the data indicating the loss of segments. Immediately after receiving this data, the sender sends the certain number of the triggering data which has very small size. When these triggering data are received at the receiver side, dup ACKs corresponding to the triggering data is replied to the sender. Then at the sending side, the dropped segment is retransmitted since a certain number of dup ACK have been received.

Third, the application can reply TCP ACK immediately by returning data although Delayed ACK is enabled. We call this method as Return method. TCP delays replying ACK as we mentioned before. TCP has the piggy back ACK function in which ACK can be replied when there is a segment to be sent at the receiver side. In Return method, the receiver replies data immediately after receiving data to reply ACK immediately.

5 Experimental Setup

In this section, we explain the experimental environment and settings. Figure 5 shows the experiment network we used. The network consists of three home networks illustrated at the left of the figure and one server's network illustrated at the right bottom. These four networks are connected with each other over the commercial fixed network via FTTH. The sender application runs on the Android device in the sender's network. Two receiver applications run on the Android device in each two receiver's network. Two receiver's networks are different in RSSI (Received Signal Strength Indication). Higher one is -40 dbm and lower is -58 dbm. Android devices are connected to wireless access point over IEEE802.11n. In server's network, relay server is connected to the wired router. The sender application send data to relay server and relay server relays the data to one of receiver applications. Our Methods are controlled between sender application-to-relay server and relay server-to-receiver application respectively. For the purpose of simulating the degradation of the WLAN performance, Windows PC which is the same model to relay server connected to the access point over IEEE802.11n in sender's network and can make bulk file transfer flow from a network attached storage in the same WLAN. The specification of Android

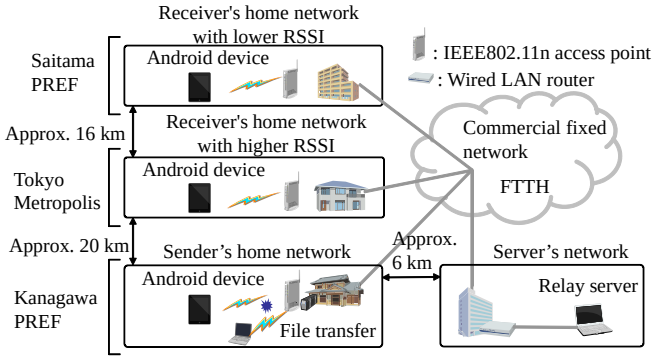

Fig. 2. The experiment network

Table 1. Device specification

	Android device	Relay server
Product name	SGH-N023[6]	VY14AC-W[7]
OS	Android OS 2.3	Windows XP SP3
CPU	S5PC110 1GHz	core2duo 1.4GHz
RAM	0.51 GB	2.96 GB
NIC	Wireless (IEEE802.11n)	Wired (1000BASE-T)

device and relay server is shown on table 1. We use AtermWR8750N as WLAN access points and wired router. Its available bandwidth is about 184 Mbps as an wireless access point and about 840 Mbps as a wired router[5].

In our experiment, we use the data which is generated sequentially by sliding a finger on the touch panel as the data communicated between applications on Android devices. These data are sent at 60 times by each 30ms. Each data is the size of 50 byte and is sent immediately from application to TCP. Like most of delay sensitive applications, we disable Nagle's algorithm[1] in this paper. We refer to the stream as the data generated by one sliding event and the delay of a stream as the delay calculated by the average of the delay of data of which one stream consists of. In the following part of this paper, we show the delay of the stream not the delay of each data. It is required that the delay of the real time application is less than 100ms[8]. Our goal is fulfilling this requirement. Typical statistic used to evaluate the delay is mean, median or mode. For the purpose to evaluate the delay sensitive real time application, these statistics are not considered appropriate. This is because the long delay which occur not so often spoils the real time application. So we use the 95th percentile which is the most commonly used value of percentile.

We measured to ensure that the file transfer flow degrade the performance of WLAN. Figure 3 shows the distribution of the frequencies of the delay of each 300 streams measured in the two cases whether there is file transfer flow or no.

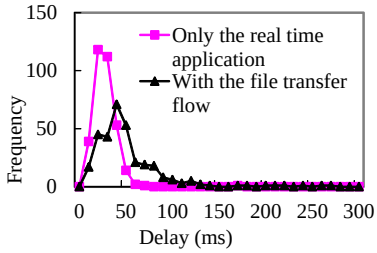

Fig. 3. The distribution of the frequencies of the delay

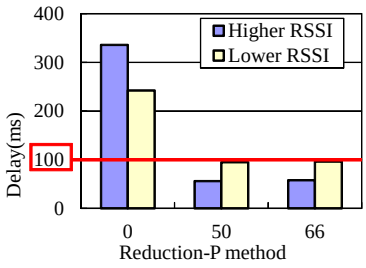

Fig. 4. The delay using TCP and Reduction-P method.

This distribution is drew with the delay of the stream on the abscissa axis and the frequencies of the delay on the ordinate. We can see that the delay in the case with the file transfer flow is longer than that in the case of only the real time application flow. In the case there is only the real time application flow, the maximum delay is 162ms and the 95th percentile of the delay is 42ms. In the case with the file transfer flow, the maximum delay is 2451ms which is not displayed and the 95th percentile of the delay is 354ms. So it is clear that the performance of WLAN is degraded due to the file transfer flow.

In our experiment, each of our methods is set as follows. In the Reduction-P method, P is set to 50 or 66. Intervals are expanded from 30ms to 60ms or 90ms respectively. In Trigger method, we set the time which is used by the receiver to infer the loss of segments to the time made by adding the interval decided by the Reduction-P method and 20ms. In the Return method, we tested whether using Return method or not.

6 Experimental Results

In this section, we show experimental results of each methods compared with results of TCP. And then, we show results of combinations of our methods. In the following section, we refer the 95th delay as the 95th percentile of the delay and figures are drew with the name of methods on the abscissa axis and the 95th delay on the ordinate. Each 95th delay in this section is calculated by delays of 300 streams respectively.

Figure 4 shows the 95th delay when using TCP and Reduction-P method with higher RSSI and lower RSSI. In the case with higher RSSI, the 95th delay by TCP is 336ms and the delay by using Reduction-50 and Reduction-33 are 56ms and 58ms respectively. In the case with lower RSSI, the 95th delay by TCP is 242ms and the delay by reducing the number of data to 30 and 20 are 95ms and 98ms respectively. By reducing the number of data to 30, the delay is reduced 83%. Although between 30 and 20 as the number of data, there is just a marginal difference. Figure 5 shows the 95th delay when using TCP and Trigger method with higher RSSI and lower RSSI. In the case with higher RSSI,

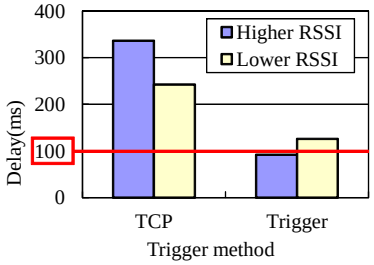

Fig. 5. The delay when using TCP and Trigger method.

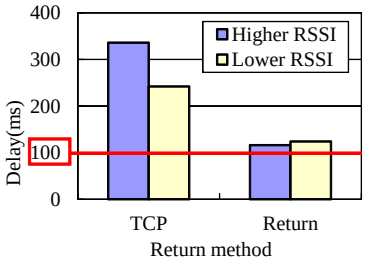

Fig. 6. The delay when using TCP and Return method.

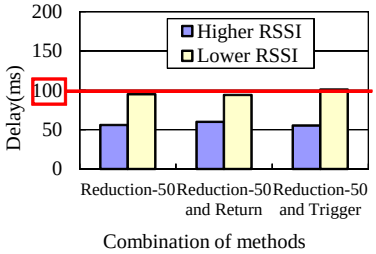

Fig. 7. The delay when using only Reduction-50, by Reduction-50 with Trigger and Reduction-50 with Return.

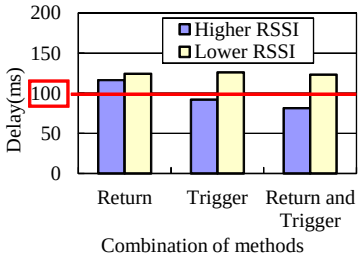

Fig. 8. The delay when using only Trigger, by only Return and Trigger with Return.

the 95th delay by using Trigger method is 96ms and is reduced 73% compared with the delay using TCP. In the case with lower RSSI, the 95th delay by using Trigger method is 126ms and is reduced 48%. Figure 6 shows the 95th delay when using TCP and Return method with higher RSSI and lower RSSI. In the case with higher RSSI, the 95th delay by using Return method is 116ms and is reduced 65% compared with the delay using TCP. In the case with lower RSSI, the 95th delay by using Return method is 124ms and is reduced 49%.

Figure 7 shows the 95th delay with higher RSSI and lower RSSI when using only Reduction-50, by Reduction-50 with Trigger and Reduction-50 with Return. In the case with higher RSSI, the 95th delay by using Reduction-50 with Trigger and with Return are 55ms and 60ms respectively and is reduced as much as 84% compared with the delay using TCP. In the case with lower RSSI, the 95th delay by using Reduction-50 with Trigger and with Return are 101ms and 94ms respectively and is reduced as much as 61%. It is needless for Reduction-50 method to be in combination with other methods. In the case with higher RSSI, Reduction-50 and Trigger decrease their gains in combination. In the case with lower RSSI, Reduction-50 and Return too.

Figure 8 shows the 95th delay with higher RSSI and lower RSSI when using only Trigger method, by only Return method and Trigger method with Return method. In the case with higher RSSI, the 95th delay by using Trigger and Re-

turn at the same time is 81ms and is reduced 76%. In the case with lower RSSI, the 95th delay by using Trigger and Return at the same time is 123ms and is reduced 49%. In the case with higher RSSI, Trigger and Return increase their gains in combination. Although in the case with lower RSSI, the gain is marginal. We also have concern about overhead of our method. Indeed the consumption of bandwidth by using Trigger and Return at the same grows about three times as much as only user operation data only. But the overhead is negligible, because overhead bandwidth up to 100Kbps is enough narrow than the available bandwidth up to 180Mbps[5].

7 CONCLUSION

We have ensured that the file transfer flow degrade the performance of WLAN and shown that our methods running on application can reduce the 95th percentile of the delay as much as 84% to less than 100ms in the case that the performance of WLAN is degraded by the file transfer flow. Though the method reducing the number of data is always the most effective, the method trying to trigger TCP fast retransmission is more effective in the case with higher RSSI and less effective in the case with lower RSSI than the method returning ACK immediately. This is because the delay of data is stable in the case with higher RSSI though it is moving strenuously in the case with lower RSSI. The method which tries to trigger the TCP fast retransmission earlier can reduce the delay due to losses of segments which are happened not so often in the case with higher RSSI. But this method is not so effective due to the fixed timer when the delay is changing strenuously. So the method which tries to return ACK immediately is more effective in the case with lower RSSI.

References

1. John Nagle. “Congestion Control in IP/TCP Internetworks.” RFC 896, January 1984.
2. Youngseok Lee, Kyoungae Kim and Yanghee Choi. “Optimization of AP Placement and Channel Assignment in Wireless LANs.” Local Computer Networks, 2002. Proceedings. LCN 2002. 27th Annual IEEE Conference, 6-8 Nov. 2002.
3. Jihoon Choi, Kyubum Lee, Sae Rom Lee and Jay (Jongtae) Ihm. “Channel Selection for IEEE 802.11 Based Wireless LANs Using 2.4GHz Band.” IEICE Electronics Express (ELEX), Vol. 8 (2011) No. 16 P 1275-1280, 2012.
4. Eli Brosh, Salman Abdul Baset, Dan Rubenstein and Henning Schulzrinne. “The Delay-Friendliness of TCP.” In SIGMETRICS, June, 2008.
5. NEC Corporation. Product Information of AtermWR8750N. [Online]. Available: http://121ware.com/product/ atermstation/product/warpstar/wr8750n-hp/.
6. SAMSUNG. Product Information of GALAXY Tab SC-01C. [Online]. Available: http://www.samsung.com/jp/support/model/SGH-N023CWNDCM.
7. NEC Corporation. Product Information of VersaPro. [Online]. Available: http://121ware.com/e-manual/m/nx/vp/base/vy14acw.html.
8. ITU-T Recommendation Y.1541 : Network performance objectives for IP-based services. 2011.