TCP HandOff: A Practical TCP Enhancement for Heterogeneous Mobile Environments

Xiuchao Wu, Mun Choon Chan and A. L. Ananda
School of Computing, National University of Singapore
#3, Science Drive 2, Singapore 117543
Email:{wuxiucha, chanmc, ananda}@comp.nus.edu.sg

Abstract— **In recent years, many different kinds of wireless access networks have been deployed for the Internet and have become inseparable parts of the Internet. But TCP, the most widely used transport protocol of the Internet, was designed for stationery hosts. In particular, TCP faces severe challenges when user moves around in these networks and handoff occurs frequently.**

In this paper, TCP HandOff (TCP-HO), a practical end-to-end mechanism, is proposed for improving TCP performance in heterogeneous mobile environments. TCP-HO assumes that a mobile host is able to detect the completion of handoff immediately and has a coarse estimation of new wireless link's bandwidth. When a mobile host detects handoff completion, it will immediately notify the server through *two* duplicate ACKs, whose TCP option also carries the bandwidth of new wireless link. After receiving this notification, the server begins to transmit immediately and keeps updating $ssthresh$ according to the bandwidth from mobile host and its new RTT samples. This updating will be stopped after *four* RTT samples or after congestion is detected.

TCP-HO has been implemented in FreeBSD 5.4. Experimental results show that TCP-HO does improve TCP performance without adversely affecting cross traffic in a heterogeneous mobile environment.

I. Introduction

In recent years, many kinds of wireless networks, such as cellular network (WCDMA[1]), Wireless LAN (Wi-Fi[2]), and Wireless MAN (WiMax [3]), have been deployed and have become integral parts of the Internet. These networks complement each other in terms of coverage, bandwidth, latency, etc. and form a heterogeneous mobile environment. These wireless networks along with portable and affordable computing devices, such as laptops and smart phones, enable the wide spread and affordable mobile Internet access.

But TCP [4], the most widely used transport protocol of the Internet, was designed for stationery hosts. When a user moves around in these heterogeneous networks, TCP performance is very poor due to the challenges brought by handoff.

Within the current heterogeneous mobile environments, many kinds of handoff may occur and they can be classified in many different ways. For example, in [5], handoff is classified into vertical handoff and horizontal handoff according to the techniques used by the wireless networks. In this paper, we classify handoff from TCP point of view. More specifically, since the BDP (Bandwidth-Delay Product) affects TCP performance significantly, we classify handoff according to BDP of old network path (before handoff) and that of new network path (after handoff).

Handoff is first classified into HH (horizontal handoff) and VH (vertical handoff) based on whether BDP changes significantly during handoff. Within HH, BDP of new and old paths do not vary appreciably. For VH, the difference in BDP is large. According to the value of BDP, HH is further divided into L-HH and H-HH. Within L-HH, BDPs of both paths are low. Within H-HH, BDPs of both paths are high. According to the direction of BDP change, VH is further divided into D-VH (Downward-VH) and U-VH (Upward-VH) [6]. During all kinds of handoff, there is normally a long disconnection time, which means a period of *zero* BDP. Figure 1 shows BDP fingerprint of our four kinds of handoff.

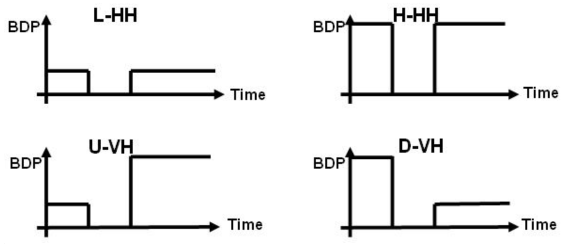


Fig. 1. BDP Fingerprint of Different Kinds of Handoff

Although TCP reacts to changes of network capacity through congestion control, large BDP changes during handoff can still bring severe challenges. In this paper, TCP-HO, a practical end-to-end mechanism, is proposed for solving the challenges brought by all different kinds of handoff with the aim to improve TCP performance in heterogeneous mobile environments.

The rest of this paper is organized as follows. In section II, the challenges faced by TCP during each kind of handoff are analyzed. Section III presents related work, and the details of our proposal, TCP-HO, are presented in section IV. The testbed and experimental results are described in section V, and conclusions are drawn in Section VI.

II. TCP During Handoff

TCP, the most widely used transport protocol, uses congestion control to probe network capacity and keep network stability. Within TCP congestion control [7], TCP sender maintains two variables, $cwnd$ (which determines the current sending rate) and $ssthresh$ (which is a coarse estimation of the network path's BDP).

When $cwnd$ is less than $ssthresh$, the sender is in slow start state. For each new acknowledgement, $cwnd$ is increased by one segment (exponential increase) so that TCP sender can quickly probe network capacity. When $cwnd$ is larger than $ssthresh$, the sender is in congestion avoidance state. And for each RTT (Round Trip Time), $cwnd$ is increased by one segment (linear increase) so that the sender can still probe network capacity and avoid frequent congestion.

TCP regards segment loss as the signal of network congestion. When congestion is detected, $ssthresh$ is reduced to half of the current $cwnd$. $cwnd$ is reduced to one if congestion is detected by timeout of retransmit timer. If congestion is detected by $3DupACK$ (three duplicate acknowledgement), $cwnd$ is reduced to half of the current $cwnd$. When a segment is lost and has to be retransmitted more than once, TCP retransmission timer adopts an exponential back-off algorithm with a maximal *timeout* value (64 seconds).

With TCP congestion control and its ACK-based self-clock, TCP sender can react to slow changes of network capacity and achieve good throughput. However, during handoff, BDP changes abruptly and long disconnection destroys TCP's self-clock. In the following paragraphs, we will analyze the problems faced by TCP during all kinds of handoff.

First, unless seamless handoff is implemented, *many segments are lost during handoff*. However, for heterogeneous networks, availability of seamless handoff is unlikely due to its complexity. As a result, when handoff occurs, segments will be discarded at the base station of previous wireless link.

Second, *after handoff completion, TCP sender still sillily waits for timeout* because TCP sender does not know the completion of handoff. Due to TCP's exponential retransmission timer back-off and long disconnection time of handoff, the sender may wait for quite a long time and precious bandwidth of new wireless link may not be utilized.

Third, *after handoff completion, the sending rate is quite slow for a long time*. During handoff, timeout occurs at TCP sender, $ssthresh$ is reduced to half of the $cwnd$, and $cwnd$ is reduced to one. TCP sender needs some time to fully utilize the bandwidth of new link. Especially when $ssthresh$ is much less than BDP of new path, TCP sender will enter into congestion avoidance state too early. This problem is worse in H-HH than in L-HH, and it is much worse in U-VH because the BDP of new path is much larger than $ssthresh$, a coarse estimation of old path's BDP.

Fourth, *TCP sender may over-shoot the new wireless link after completion of D-VH*. Within D-VH, the BDP of new path is much less than $ssthresh$. The exponential $cwnd$ increase in slow start state may cause multiple segment loss, trigger timeout, and adversely affect TCP throughput.

Table I summarizes the problems faced by TCP during our four kinds of handoff that may occur in the current heterogeneous mobile environments.

	Segment Loss	*Silly Waiting*	*Slow Start*	*Over-shooting*
L-HH	yes	yes	a little	no
H-HH	yes	yes	severe	no
U-VH	yes	yes	very severe	no
D-VH	yes	yes	very little	yes

TABLE I
PROBLEMS BROUGHT BY DIFFERENT KINDS OF HANDOFF

III. RELATED WORK

The root of TCP's poor performance during handoff is that congestion control is carried out by the server and handoff is only known to mobile host. Based on this understanding, we classify the related works according to where adaptation takes place.

1) Network-Centric Approaches: I-TCP [8] and MTCP [9] split a connection between a server and a mobile host at the base station. Handoff is entirely handled by mobile host and base stations. TCP server continues to send segments to the old base station during a handoff. These segments will be buffered at the old base station and forwarded to the new base station after handoff is completed. In this way, handoff is hidden from TCP servers. But within heterogeneous mobile environments, wireless networks may use different techniques and belong to different administrative domains. These proposals may not be practical again. In addition, base stations need to maintain large buffers within high-speed wireless networks.

M-TCP [10] only works at the base station. Base station holds back the ACK of the last byte. When it detects handoff occurrence, it sends back the last byte's ACK with *zero window*,which will force TCP sender to enter into persist mode, thus the values of $cwnd$ and $ssthresh$ are frozen. When a new base station detects handoff completion, it immediately notifies TCP sender to resume transmission with the frozen $cwnd$ and $ssthresh$. M-TCP is a good network-centric solution. But the overhead of base station is too large and increases with the speed of wireless networks. More severely, M-TCP, I-TCP, and MTCP can not work with IPSEC [11].

2) Receiver-Centric Approaches: RCP [12], which moves congestion control to mobile host, is the most radical proposal of this category. Since mobile host knows handoff and carries out congestion control now, it can solve all problems brought by handoff. But it is too dangerous to let mobile users decide the sending rate of fixed servers, especially in today's Internet. Freeze-TCP [13] and TCP ACK-Pacing [6] are two other receiver-centric proposals. They change mobile host's behaviors during handoff with the aim to drive TCP server's congestion control for probing network capacity well.

Freeze-TCP shifts the tasks of M-TCP's base station to mobile host and avoids hold the ACK of the last byte. Freeze-

TCP assumes accurate handoff prediction so that *zero window* can arrive the server and old wireless link will not be under-utilized. But it is very hard to accurately predict handoff in heterogeneous wireless networks. In addition, after handoff completion, the server begins to transmit with previous $cwnd$ and $ssthresh$. This mechanism may work well during L-HH. Considering the faster and faster wireless networks, it may generate large data burst. Freeze-TCP does not consider the challenges brought by D-VH and U-VH too.

TCP ACK-Pacing considers the challenges brought by U-VH and D-VH. It assumes that a mobile host knows RTT of the new path and the bandwidth of the new wireless link. Hence it knows the BDP of the new path when the wireless link is the bottleneck. The ACK-generation algorithm of the mobile host is changed in order to drive the server converge to new path's BDP quickly. However, TCP ACK-Pacing can not work at all if TCP Byte-Counting [14] is used by the server. In addition, less ACKs after D-VH may generate burst data and break TCP's self clock. More ACKs after U-VH may consume precious uplink bandwidth of asymmetric wireless link. It is also possible for misbehavior users to get unfairly high throughput.

Freeze-TCP and TCP ACK-Pacing are *receiver-centric* end-to end mechanisms. They can work with IPSEC and provide good deployability since only mobile host is changed. However, their adaptations are only performed by mobile devices and their performance improvement may be constrained.

TCP-HO improves TCP performance better by bringing explicit cooperation between TCP server and mobile host. Hence, TCP-HO is a *sender+receiver approach*. Considering that more and more users are accessing the Internet through wireless networks, it should be worthwhile to change both TCP end-points.

IV. TCP HandOff Mechanism

In this section, TCP HandOff (TCP-HO), a practical end-to-end TCP enhancement for heterogeneous mobile environments, is presented. Within TCP-HO, a mobile host reports handoff information to the server. And the server is responsible to well utilize wireless links based on mobile host's feedback.

The ideal solution is to let TCP sender stop transmitting one RTT before handoff occurrence and immediately begin to transmit ($cwnd = BDP$ of new path) after handoff completion. By this way, TCP sender can fully utilize the old and new wireless links and avoid any segment loss. However, this is not a practical solution. Below is the design principles and mechanism details of TCP-HO.

A. Design Principles

1) Assumptions made must be reasonable. Since it is very hard to accurately predict handoff in heterogeneous wireless networks, mobile host of TCP-HO does not predict handoff and notify the server. Hence, TCP-HO can not avoid segment loss.

2) The mechanism should be deployable on the current Internet. In order to avoid complicating network infrastructure and work with IPSEC, TCP-HO should be a truly end-to-end mechanism. In addition, it should be easy to implement TCP-HO in the existing TCP codes.

3) The mechanism should not bring new security problems. Especially, when TCP sender accepts the notification of new link's bandwidth, it must use this information scrupulously and discard it quickly.

4) The mechanism should be friendly to cross traffic. With the increase of mobile users and the increase of wireless link's bandwidth, TCP-HO must consider its effects to network stability and cross traffic. After handoff completion, instead of sending with full speed, TCP-HO must probe network capacity as same as normal TCP. In order to accelerate capacity probing, $ssthresh$ can be set according to BDP of new path.

5) The server should perform most of the work. Considering that mobile host is usually resource constraint, most work of TCP-HO should be carried out by the server.

B. Details of TCP-HO

There are only two assumptions in TCP-HO. Firstly, a mobile host can immediately know the completion of handoff. This knowledge can be acquired easily by some cross-layer implementations. Secondly, a mobile host has a $coarse$ estimation of new wireless link's bandwidth. This assumption is reasonable since mobile host can estimate bandwidth based on the type of wireless interface used or through some bandwidth estimation mechanisms, such as [15].

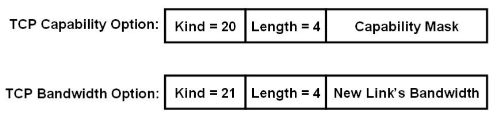


Fig. 2. TCP Options for TCP-HO

TCP-HO extends TCP by including two additional TCP options (see Figure 2). TCP Capability Option is an enabling option used in SYN segments. Each bit of its 16-bits capability mask can be used to negotiate one TCP capability. TCP-HO is negotiated through the last bit. TCP Capability Option is designed to save precious option space of SYN segment. As for TCP Bandwidth Option, it is used by mobile host to notify the server about the completion of handoff and the bandwidth of new wireless link. The unit of bandwidth is Kbps.

Within TCP-HO, when a mobile host gets to know the completion of handoff, it immediately sends out *two* duplicate ACKs with TCP Bandwidth Option which carries BWD_{new} (the bandwidth of new wireless link). Two duplicate ACKs make the notification more robust in lossy network. If more than two duplicate ACKs are sent out, they may trigger TCP sender's fast retransmit and fast recovery [7].

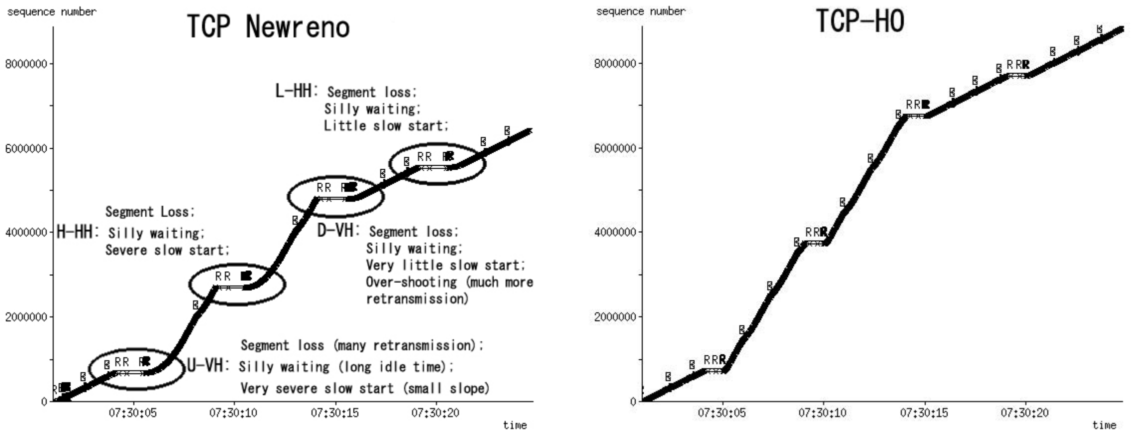
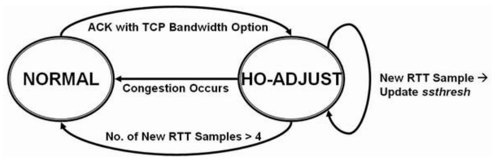


Fig. 3. Time *vs.* Sequence No. Graph of TCP Newreno and TCP-HO under our four kinds of handoff occurred in the same mobile scenario

When the server receives one ACK with TCP Bandwidth Option, it sets $ssthresh = BWD_{new} * SRTT_{old}$ and enters into HO-ADJUST state (see Figure 4). In HO-ADJUST state, for each new RTT sample, the server keeps updating $ssthresh$ according to BWD_{new} and the smooth average of new RTT samples. After congestion is detected, the server returns to normal state and works as normal TCP. According to source codes of FreeBSD 5.4 [16], four RTT samples can give a quite accurate estimation of average RTT. TCP-HO server returns to normal state after four new RTT samples. Figure 4 shows the state transition diagram used by TCP-HO sender.



ACK with TCP Bandwidth Option
NORMAL
HO-ADJUST
Congestion Occurs
New RTT Sample → Update ssthresh
No. of New RTT Samples > 4

Fig. 4. State Transition Diagram of TCP-HO Sender

Figure 3 shows the time *vs.* sequence no. graph of TCP Newreno [17] and TCP-HO during our four kinds of handoff which occur in the same mobile scenario. For better comparison, we implemented TCP-HO in NS2 [18] and generated the two graphs. This figure shows that TCP Newreno does suffer the problems pointed out by us in Table I. It also shows that TCP-HO solves all problems except segment loss and achieves much higher throughput than TCP Newreno.

Within [19], a similar idea was presented and evaluated by simulation. But its network scenario is different with ours. The authors assume that the mobile host is the sender of data.

V. Performance Evaluation

TCP-HO has been implemented in FreeBSD 5.4. For the purpose of performance evaluation, a controlled network environment is set up and many experiments are carried out.

A. Testbed Setup

Within the testbed shown in Figure 5, a Mobile Host is connected with Handoff Emulator by cross-cable and all other computers are connected by two virtual LANs of a 3COM Super Stack II Switch 3900.

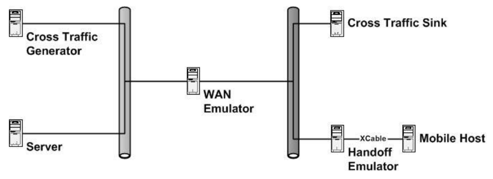
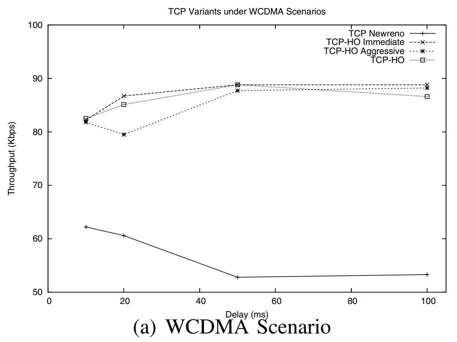


Fig. 5. Testbed

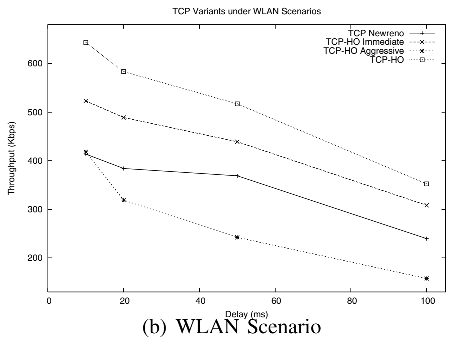
FreeBSD 5.4 is installed on all of these computers. The connection between Server and Mobile Host is the connection to be investigated. Cross traffic is generated between Cross Traffic Generator and Cross Traffic Sink in order to investigate the friendliness of TCP-HO. Iperf [20] is used by Server and Cross Traffic Generator for generating all traffics and measuring their throughput.

Dummynet [21] is used by WAN Emulator to emulate the delay and bandwidth of a wide area network. The bandwidth of WAN is set to 10Mbps. Considering the small number connections in our experiments, the queue at WAN Emulator is set to 20 packets. As for the delay of WAN, it is changed in different experiments in order to emulate that mobile host accesses servers that locate at different places.

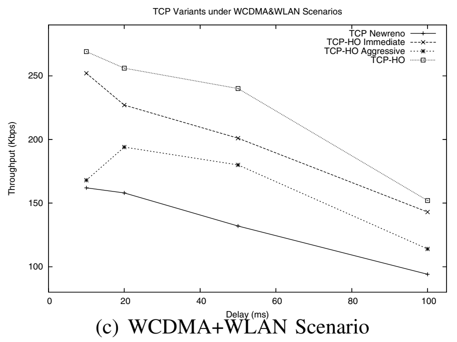
In order to better emulate the bandwidth & delay of different wireless links, instead of Mobile Host, a separate computer (Handoff Emulator) is used and its kernel is re-built with finer timer resolution. Dummynet is used for emulating mobile scenarios and a special ICMP message is used by Handoff Emulator to notify the mobile host about handoff completion and the bandwidth of new wireless link. This ICMP message emulates handoff completion detection and bandwidth estimation functions of Mobile Host.



(a) WCDMA Scenario



(b) WLAN Scenario



(c) WCDMA+WLAN Scenario

Fig. 6. Throughput between Server and Mobile Host under Different Mobile Scenarios

B. Experimental Results

In order to evaluate TCP-HO, we first generate several mobile scenarios (handoff sequences) that may occur in different mobile environments of the real world (WLAN and WCDMA). For each mobile scenario, the delay of WAN are set to different values (10ms, 20ms, 50ms, and 100ms) to create various network scenarios.

For each network scenario, the connection between Server and Mobile Host may use TCP-HO and TCP Newreno. In order to analyze the effects of different parts of TCP-HO mechanism, two additional TCP variants, TCP-HO-Immediate and TCP-HO-Aggressive, are also investigated. Within TCP-HO-Immediate, the sender immediately begins to transmit after receiving notification. But the bandwidth of new wireless link is ignored. Hence, only *immediate transmission* [22] is adopted and only the mobile host needs to be changed. In TCP-HO-Aggressive (similar to Freeze-TCP), the sender also sets $cwnd = BWD_{new} * SRTT_{old}$. Hence, for each network scenario, four experiments must be carried out.

Within each experiment, the cross traffic connection uses TCP Newreno and runs simultaneously with the connection between Server and Mobile Host. In order to get accurate results, each experiment is carried out for three times and the average is computed. In the following parts, we will present our experimental results under three emulated mobile scenarios.

1) WCDMA Scenario: In this scenario, we emulate that a user drives a car (speed: 60km/h) in a rural area covered by WCDMA network (cell coverage: 2km) for half an hour. The average connection time is 50 seconds and the disconnection time is 3 seconds. We assume that ten users share one 2Mbps WCDMA channel and the bandwidth of each user is about 200Kbps.

Figure 6(a) shows the throughput between Server and Mobile Host when different TCP mechanisms are used. This figure shows that TCP-HO and its variants do improve TCP performance. The difference between TCP-HO and its variants is very small. It is reasonable since L-HH dominates the WCDMA mobile scenario and *silly waiting* of TCP is the main problem. This results also concur with Freeze-TCP's claim that *immediate transmission* dominates Freeze-TCP's performance enhancement. We also find that the throughput does not response to the change of WAN delay well. The reason is that RTT is dominated by the slow WCDMA link whose base station has a large buffer (20 packets).

2) WLAN Scenario: In this scenario, we emulate that a user drives a car (speed: 30km/h) around a campus with some Wi-Fi hot-spots (cell coverage: 100m) for 15 minutes. The average dwelling time is 12 seconds and the disconnection time is 10 seconds (bad coverage). Per-cell user number is either large (9-10) or very small (1-2). These users share 7Mbps channel bandwidth (Effective Channel Bandwidth of IEEE 802.11b [23]). Available bandwidth for each user can vary significantly depends on the number of users per cell.

Figure 6(b) shows the throughput between Server and Mobile Host in WLAN mobile scenario. It shows that TCP-HO performs much better than Newreno. It also shows that TCP-HO-Immediate achieves trivial improvement and TCP-HO-Aggressive is even worse than Newreno. This is reasonable since H-HH, U-VH and D-VH dominate this mobile scenario. TCP-HO-Immediate does not solve severe *"slow start"* problem of H-HH and U-VH. In case of TCP-HO-Aggressive, during every kind of handoff occurred in this high-speed wireless network, the large data burst makes it immediately suffer multiple segment loss, which will trigger timeout and harm its throughput.

Prob. / Disconn. Time	*WCDMA*	*WLAN*
WCDMA	0.33 / 3s	0.67 / 5s
WLAN	0.9 / 5s	0.1 / 10s

TABLE II
HANDOFF PROBABILITY AND DISCONNECTION TIME

3) WCDMA&WLAN Scenario: In this scenario, we emulate a user who drives a car (speed: 45km/h) around a city with WCDMA coverage and sporadic Wi-Fi hot-spots for 20 minutes. If Wi-Fi hot-spot exists, the user always switches to Wi-Fi. Otherwise, WCDMA is used. The probability of handoff between two networks and the disconnection time follow the numbers in Table II. Here, we assume that two Wi-Fi hot-spots exist in one WCDMA cell.

The average dwelling time of WCDMA cell is 20 seconds

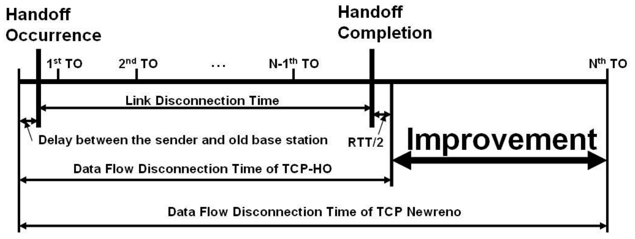
TCP Variants	WCDMA Scenarios (Mbps)				WLAN Scenarios (Mbps)				WCDMA&WLAN Scenarios (Mbps)			
	10ms	*20ms*	*50ms*	*100ms*	*10ms*	*20ms*	*50ms*	*100ms*	*10ms*	*20ms*	*50ms*	*100ms*
TCP Newreno	3.69	3.323	2.52	1.653	3.46	3.08	2.317	1.603	3.64	3.27	2.48	1.63
TCP-HO Immediate	3.65	3.287	2.527	1.633	3.33	2.99	2.26	1.583	3.54	3.21	2.48	1.63
TCP-HO Aggressive	3.66	3.297	2.537	1.623	3.38	3.09	2.273	1.563	3.56	3.19	2.38	1.57
TCP-HO	3.657	3.293	2.533	1.633	3.2	2.9	2.133	1.527	3.52	3.173	2.433	1.57

TABLE III
THROUGHPUT OF CROSS TRAFFIC FLOW WHEN IT RUNS WITH DIFFERENT TCP VARIANTS

and per-user bandwidth is 200Kbps. The average dwelling time of Wi-Fi is 8 seconds and per-cell user number follow the same distribution used by WLAN scenario.

Figure 6(c) shows the throughput between Server and Mobile Host in WCDMA&WLAN mobile scenario. It looks like a mixture of WCDMA scenario and WLAN scenario. It is reasonable since S-HH, H-HH, U-VH, and D-VH all occur in this scenario. This figure also indicates that TCP-HO does help WCDMA users to utilize Wi-Fi hot-spots.

According to the above results, TCP-HO is the best mechanism for improving mobile host's throughput in most of our experiments. Table III shows the throughput of cross traffic flow under different mobile scenarios when it runs with different TCP versions. It indicates that *TCP-HO does improve TCP performance of Mobile Host without adversely affecting cross traffic*. The burstiness of TCP-HO-Aggressive does not hurt the cross traffic in most of our experiments. The reason may be that the burstiness hurts itself too much and makes it generate less traffic. Hence, the cross traffic acquires more bandwidth of the Wide Area Network. Additional experiments are needed to investigate this issue further by using more mobile users and cross traffics.



Handoff Occurrence
Handoff Completion
1st TO
2nd TO
...
N-1th TO
Nth TO
Link Disconnection Time
Delay between the sender and old base station
RTT/2
Improvement
Data Flow Disconnection Time of TCP-HO
Data Flow Disconnection Time of TCP Newreno

Fig. 7. Improvement of TCP-HO on Data Flow Disconnection Time

The improvement of TCP-HO's *immediate transmission* on the disconnection time of data flow can be analyzed as follow. If link disconnection time is so long that the *timeout* of retransmission timer reaches 64 seconds, the improvement is between 0 to 64 seconds. Otherwise, the improvement can be presented by Figure 7. With the assumption that handoff may be completed at any time between $N-1^{th}$ and N^{th} expiration of retransmission timer, through some calculous operations, we deduce that the improvement is about $(2\ln 2 - 1)$ of the link disconnection time.

VI. CONCLUSION

In this paper, TCP-HO, the first practical end-to-end TCP enhancement, is proposed to solve the challenges brought by all kinds of handoff that may occur in heterogeneous mobile environments. Experimental results show that in heterogeneous mobile environments, TCP-HO improves TCP performance a lot without adversely affecting cross traffic. These results indicate that TCP-HO is really a promising solution.

REFERENCES

[1] "3gpp," available online at http://www.3gpp.org.
[2] *Wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, IEEE P802.11, IEEE Std., 1999.
[3] "Wimax," available online at http://www.wimaxforum.org.
[4] J. Postel, "Transmission control protocol - darpa internet program protocol specification," RFC 793, DARPA, Sept. 1981.
[5] S. Mark, "Vertical handoffs in wireless overlay networks," Technical Report CSD-96-903, UCB, Tech. Rep., 1996.
[6] Y. Matsushita, T. Matsuda, and M. Yamamoto, "Tcp congestion control with ack-pacing for vertical handoff," in *IEEE WCNC*, 2005.
[7] V. Jacobson, "Congestion avoidance and control," in *ACM SIGCOMM*, 1988.
[8] A. Bakre, "I-tcp: indirect tcp for mobile hosts," in *ICDCS*, 1995.
[9] R. Yavatkar and N. Bhagawat, "Improving end-to-end performance of tcp over mobile intemetworks," in *IEEE Worhhop on Mobile Computing Systems and Applications*, 1994.
[10] K. Brown and S. Singh, "M-tcp: Tcp for mobile cellular networks," in *ACM CCR*, 1997.
[11] S. Kent and R. Atkinson, "Security architecture for the internet protocol," RFC 2401, Nov. 1998.
[12] H. Hsieh, K. Kim, Y. Zhu, and R. Sivakumar, "A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces," in *ACM MOBICOM*, Sept. 2003.
[13] T. Goff, J. Moronski, D. S. Phatak, and V. Gupta, "Freeze-tcp: A true end-to-end tcp enhancement mechanism for mobile environments," in *IEEE INFOCOM*, 2000.
[14] M. Allman, "Tcp congestion control with appropriate byte counting (abc)," RFC 3465, Feb. 2003.
[15] X. Wu and A. L. Ananda, "Link characteristics estimation for ieee 802.11 dcf based wlan," in *IEEE LCN*, 2004.
[16] "Freebsd," available online at http://www.freebsd.org.
[17] S. Floyd, T. Henderson, and A. Gurtov, "The newreno modification to tcp's fast recovery algorithm," RFC 3782, Apr. 2004.
[18] "Ns2 network simulator," http://www.isi.edu/nsnam/ns/.
[19] K. Tsukamoto, Y. Fukuda, Y. Hori, and Y. Oie, "New flow control schemes of tcp for multimodal mobile hosts," in *IEEE VTC-Spring*, 2003.
[20] "Iperf," available online at http://dast.nlanr.net/Projects/Iperf/.
[21] L. Rizzo, "Dummynet: a simple approach to the evaluation of network protocols," *ACM CCR*, vol. 27, no. 1, pp. 31–41, 1997.
[22] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE JSAC Special Issue on Mobile Computing Network*, 1994.
[23] *Wireless LAN medium access control (MAC) and physical layer (PHY) specifications—higher-speed physical layer extension in the 2.4GHz band*, IEEE P802.11, IEEE Std., 1999.