Cross-Layer Assisted TCP for Seamless Handoff in Heterogeneous Mobile Wireless Systems

Hassan Sinky and Bechir Hamdaoui
School of Electrical Engineering and Computer Science, Oregon State University
{sinkyh,hamdaoub}@onid.orst.edu

Abstract— **We propose cross-layer assisted TCP enhancements to improve service quality of mobile users during handoff. Specifically, we investigate and resolve TCP issues that pertain to handovers between wireless networks with varying data rates. New adjustments to TCP's round trip time, congestion window, and acknowledgment mechanism are proposed to ensure seamless handoff. The performances of the proposed cross-layer assisted handoff TCP (TCP-CLAH) are evaluated and compared to those of TCP NewReno (TCP-NR). Our results show that TCP-CLAH outperforms TCP-NR significantly in terms of jitter, round-trip time, and queue size, thereby reducing service disruptions and glitches of ongoing communications substantially during a handoff.**

I. Introduction

The convenient features of wireless technology, such as mobility, portability, and ease of use and deployment, are the main reasons behind the technology's tremendous success. Heterogeneous wireless systems have been designed to take advantage of these wireless features. We as users and consumers generally take service availability for granted. We usually don't worry or think about it because we assume we will always have it. Not only have we taken for granted service availability but also the quality of that service. Even the smallest disruption or glitch in any of these technologies would be considered an unbearable inconvenience. As the number of mobile users increase so does the demand for top notch performance and service quality. Due to this growing demand for access to communication services anywhere and any time an accelerated development towards the integration of various wireless access technologies has resulted.

Heterogeneous wireless systems provide coexisting technologies resulting in more services and higher data rates while also providing a global roaming/ always connected environment through a diverse range of mobile access networks. Thus, within a heterogeneous wireless system a mobile device may encounter many networks with varying data rates and performance, sometime even drastic differences. A typical heterogeneous networking scenario involves mobile terminals with multiple network interfaces which are capable of choosing the best possible network or link for data. For instance selecting a wifi network over a cellular network when you are in range to save battery and achieve higher data rates. Another example would be having a Skype conversation on a mobile device and seamlessly, gracefully and intelligently select networks in range without the user feeling any disruption during the conversation. This type of graceful network selection is generally referred to as a seamless handover or handoff.

Drawbacks and issues begin to arise when handovers occur between networks with varying capacities and data rates such as between Wifi and cellular networks. In this paper we analyze TCP issues due to network handovers and propose a cross-layer assisted handoff TCP (TCP-CLAH) to improve overall performance by minimizing user perceived glitches. In this work, TCP NewReno (TCP-NR) is compared with our TCP-CLAH scheme and results are evaluated. Our measurements show that TCP-CLAH outperforms TCP-NR significantly in terms of jitter, round-trip time, and queue size, yielding much more robust, reliable, handoff resistant and efficient data transfers.

In our research, a cross-layer assisted handoff TCP is proposed in order to alleviate performance issues and glitches that occur during a handoff between networks of varying performance and data rates. Our contributions in this work are:

- Performance evaluation and analysis of TCP NewReno during handovers in heterogeneous networks.
- Cross-layer design and adjustment of TCP parameters to improve service quality during handovers.
- Establishing that reducing congestion window at time of handoff is essential for ensuring service continuity and minimizing disruption and glitches.
- To our knowledge, this work is the first to investigate handoff connection disruptions and glitches with TCP NewReno, and to improve service continuity during handovers through BDP-based congestion window adjustment.

The rest of the paper is organized as follows. We begin by discussing related work in Section II. TCP issues as a result of network handovers are discussed in Section III. We then, in Section IV, discuss the bandwidth delay product performance metric as well as propose a scheme to improve TCP during a handoff. In Section V, we introduce our test topology and perform a series of tests to evaluate and compare the performance of our proposed TCP-CLAH with TCP-NR. Finally, we conclude the paper in Section VI.

II. Related Work

Network handovers have been a well researched subject. However, researchers today have focused mainly on the

978-1-4799-1353-4/13/$31.00 ©2013 IEEE

handoff detection and decision process where little has been done to alleviate transport protocol issues resulting from a change in network point of access especially between networks with varying data rates.

In [1] the authors divide a handoff into three phases. The first phase is when TCP goodput increases, the second phase is when TCP goodput decreases and finally a TCP timeout happens and the sending rate slowly increases. TCP is adjusted based on the phase, user's speed, location and initial congestion window. The authors mostly focus on the physical aspect of the mobile user and handoff decision rather than the transport protocol itself.

In [2] Freeze TCP (ftcp) when a pending handover is to occur a mobile user sends a zero window advertisement (ZWA) to the sender. The sender then freezes its timers and suspends all transmissions. Once the handover is complete the mobile user sends three successive acknowledgments to the sender to resume transmission. The time before a disconnection occurs is referred to as the warning period. This period is difficult to predict and if predicted too early the sender enters persist mode too early, thereby increasing idle time and decreasing throughput. If predicted too late, the sender may not receive the ZWA in time, thereby causing the congestion window to reduce due to lost packets and in turn decrease throughput.

In [3], [4] the authors have designed a cross-layer assisted TCP solution for handoff situations. It is shown that TCP can benefit greatly by adjusting parameters according to a network handoff. When a handoff occurs RTT and RTO estimations are reset only until all packets sent before the handoff have been acknowledged. A reduction in congestion window is done only when needed. In addition, the slow start threshold is set to the congestion window which unnecessarily places TCP in the congestion avoidance phase when a handoff occurs. Unlike these works, we propose a different adjustment to TCP based on the path bandwidth delay product as well as a different principle in setting the RTT and RTO estimations. Finally, our proposed scheme uses TCP NewReno whereas authors in these works use TCP SACK.

In the next section we discuss network handovers and their effect on TCP.

III. TCP Handoff Issues

Handovers can be a result of preference, service availability, network quality or forced. As mentioned earlier, researchers have mainly focused on the handoff decision and detection process whereas our work focuses on how to improve TCP in handoff scenarios and minimize connection ***disruptions*** and ***glitches***. There are two types of handovers: horizontal and vertical. A horizontal handoff takes place between points of attachment supporting the same network technology. For instance, a mobile user were walking through a department building may undergo a few horizontal handovers. On the other hand a vertical handoff occurs between points of attachment supporting different network technologies [5]. This occurs when a mobile user exits a building, out of range of Wifi, and switches to a cellular data network.

Little has been done in the area of analyzing and resolving TCP's drawbacks during a handoff. Before introducing these drawbacks, we describe two means of a handoff:

- **Break before make**: terminates the old link before the handoff completes. This causes a visible disruption in connectivity and packet losses as packets that have been sent using the old interface will have to be retransmitted.
- **Make before break**: terminates the old connection only when the new link is operable. This type allows for no packet loss due to buffering and is referred to as a soft handover.

TCP is optimized for wired networks where packet loss is assumed to be caused by congestion only [6]. However, when we move to wireless scenarios this assumption fails since wireless links experience error rates much larger than wired networks. Thus, if packets are dropped or delayed the TCP congestion control mechanism is initiated to alleviate the problem which reduces throughput drastically. As a result, during a handoff, the main dilemma we are left with is trying to help TCP avoid incorrectly initiating its congestion avoidance and slow start mechanisms. In order to successfully do this TCP **must** differentiate between packet delay caused by actual real packet loss and congestion on the current link and packet delay simply caused by vertical or horizontal handoff.

A. Handoff Induced Issues

There are different TCP problems that arise depending on the type of handoff that occurs [7]. We classify these TCP issues based on two handoff scenarios:

1) High-delay network to low-delay network:

- **Packet reordering:** This issue occurs when the source node switches to considerably faster link. Packets sent on the new link overtake packets that have been sent on the old link which causes out of order packets to arrive at the destination, resulting in duplicate acknowledgements to be sent (dupacks). This is an issue because during a handoff TCP would misinterpret 3 dupacks as being caused by congestion, loss or link error and enters congestion avoidance and retransmit packets incorrectly assumed to be lost when in reality this was caused by a handoff.
- **Inflated retransmission timeout (RTO):** Naturally, a high-delay link will have a high RTT and RTO value. Since the RTO is updated once in an RTT, when a handoff occurs, the RTO will converge slowly to the new low RTO value. This is an issue because invoking RTO recovery to recover lost packets will take a longer time due to the high RTO value. In other words, in the event of lost packets on a faster link TCP would not be able recover quickly due to the high RTO value of the high-delay link. Since after the handoff we are on a low-delay link we want TCP to be able to recover

lost packets faster using a smaller RTO representing the new faster link.

2) Low-delay network to high-delay network:

- **Spurious or false retransmission timeout (RTO):** This issue arises when we handoff to a slower link. The low-delay link before the handoff has a low RTO value. Packet acknowledgments before the handoff take the high-delay after the handoff. This causes TCP to falsely or spuriously timeout assuming packets have been lost due to a low RTO value from the low-delay link.
- **Link overshoot:** After a handoff to a low-delay link the sender may inject more data onto the link than can be handled resulting in dropped packets. Specifically when TCP is in the slow start phase the congestion window is doubled for every acknowledgement received. If a handoff to a low BDP link occurs during the slow-start phase a congestion window increase may result in dropped packets. This is referred to as a slow-start overshoot. TCP may interpret this incorrectly and reset the congestion window which is a performance hit we want to avoid.

In the next section we introduce the use of BDP as a means to evaluate network capacity. A cross-layer assisted handoff TCP scheme is then proposed to minimize handoff induced glitches and disruptions in service continuity.

IV. BDP AND TCP HANDOFF

We argue that in order to minimize connection glitches and maintain adequate service continuity during a handoff the bandwidth delay product (BDP) of the link must be considered. The BDP refers to the maximum amount of data that can be sent on a network link at any given time. In other words, it represents the amount of data that is sent before the first ACK is received [8]. The BDP is represented by the product of a link's capacity and its round-trip time. By definition then the path BDP, BDP_{path}, is the product of the path bottleneck throughput, R_{min}, and the path round-trip time, RTT_{path}[1]; i.e., $BDP_{path} = R_{min} \times RTT_{path}$. When propagation and queuing delays are neglected, we can write

$$BDP_{path} = R_{min}\left(\sum_{i=1}^{n}\frac{S}{R_i} + \sum_{i=1}^{m}\frac{S}{R'_i}\right) \quad (1)$$

where n and R_i are the number of hops and throughput on the forward path, m and R'_i are the number of hops and throughput on the backward path, and S is the packet size. As done in [8], the path BDP can be upper bounded by

$$R_{min}\left(n\frac{S}{R_{min}} + m\frac{S}{R'_{min}}\right) \quad (2)$$

where R'_{min} is the bottleneck throughput in the backward path. Now considering that the forward and backward paths are likely to be the same, the upper bound given in Equation (2) can be replaced by $S \times N$ where $N = n + m$[3]; i.e.,

$$BDP_{path} \leq S \times N \quad (3)$$

Note that having the path BDP exceed $S \times N$ results in packets being queued at the bottleneck [8]. In what follows, let $BDP_{UB} = S \times N$.

A. Design

Our goal is to alleviate issues presented in Section III by proposing modifications to TCP NewReno allowing for robust data transfers and maintain service continuity. Thus, our intention is to minimize slight **disruptions** and **glitches** immediately following a handover.

In designing our scheme two assumptions are made:

- We assume a soft handover scenario where a new connection is made before the previous connection is terminated. This allows for no packet loss due to buffering. These handovers involve scenarios where a mobile node is in range of multiple networks.
- We assume a cross-layer notification of a handoff indicating a drastic change in bandwidth. That is a notification communicated to TCP indicating a significant increase or decrease in bandwidth.

TCP-CLAH brings three modifications to TCP New Reno in order to provide a seamless handover:

- Congestion window is set to BDP_{UB} during handoff
- RTT and RTO estimations are reset upon a handoff as if for a new connection
- Duplicate acknowledgments are temporarily disabled during a handoff

A large congestion window during a handoff can be detrimental to service continuity. Maintaining a large congestion window increases the probability of contention, round-trip time, queuing and packet loss which can degrade TCP performance drastically. We argue, by setting the congestion window to the path BDP, that is the maximum number of bytes that can be in transit across the path, guarantees we are transmitting the maximum amount of bytes while avoiding packet queuing at intermediate links. As mentioned in Section III we are concerned with two handoff scenarios.

For a high-delay to low-delay network handoff we apply Algorithm 1. The intuition behind this is that if the RTO is set too high TCP may be slow to recover lost packets. If set too low, we risk spurious RTOs where TCP assumes packets are lost when in reality they may simply be on their way. Through testing we found by resetting these values TCP converges quicker to the new network's RTT and RTO estimations. In addition, duplicate acknowledgments are temporarily disabled to avoid false dupacks caused by packet reordering. Then, the congestion window is set to the path BDP. Often times this value is less than the current congestion window size. One would think that a transition from a high-delay network to low-delay network involves further increasing the congestion window to fully utilize the link. However, as mentioned earlier, injecting more data

[1]RTT consists of propagation and transmission delays only; i.e., it does include queuing delays.

than the path BDP causes packets to be queued at the bottleneck. Thus setting the congestion window to the path BDP, albeit less, is sufficient and produces better results. The achieved throughput is not compromised by this action as will be shown in Section V.

if *Notification of handoff to low-delay* **then**
  reset *RTT* and *RTO* estimations;
  temporarily disable *dupacks*;
  obtain path *BDP*;
    set *cwnd* = path BDP;
end

Algorithm 1: TCP adjustments for handoff to low-delay

For low-delay to high-delay network handovers we apply Algorithm 2. Resetting the RTT and RTO estimations to their initial states allows for TCP to relearn the new network conditions quicker and avoid spurious RTOs. Since we are moving to a slower network, the three duplicate acknowledgments are not an issue. Again setting the current congestion window to the path BDP avoids a link overshoot.

if *Notification of handoff to high-delay* **then**
  reset *RTT* and *RTO* estimations;
  obtain path *BDP*;
    set *cwnd* = path BDP;
end

Algorithm 2: TCP adjustments for handoff to high-delay

In Section V we analyze the results obtained from measuring service continuity and performance factors (i.e. jitter, round-trip time, queue size and throughput) conducted on the network topology shown in Figure 1.

V. Performance Evaluation and Results

In this section we discuss metrics measured to compare the performance and service continuity of TCP during a handoff. We then discuss the configuration of our test topology and simulation results.

A. Performance Metrics

Overall network performance is impacted by many environmental and physical aspects and limitations. It is crucial to understand what factors play a role in network performance and what can be done to improve them. We discuss and measure factors that influence network performance and service continuity during handovers. The following subsections briefly discuss these factors.

1) Jitter: Jitter, also referred to as packet delay variation (PDV), is the difference in end-to-end delay between packets. Thus, variable delay causes jitter. On the other hand a network that experiences a constant amount of latency between packets has no jitter [9]. In addition, as a result of congestion a packet may be queued or delayed on a path where there was no queuing or delay for other

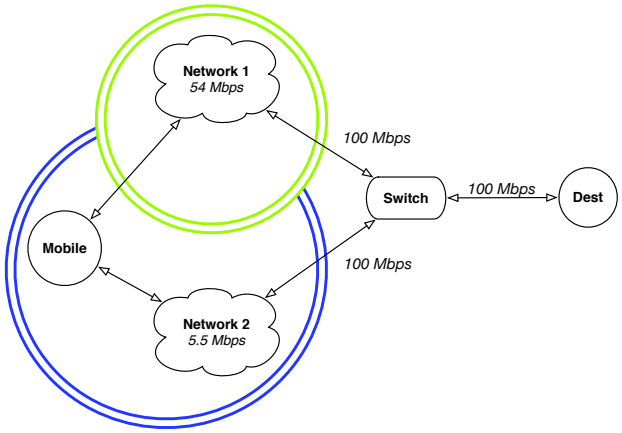

Fig. 1: Test Topology Configuration

packets [10]. This in turn causes a variation in latency. In multimedia applications such as VoIP, high variation in packet delay can result in poor communication quality where audio/video delays and dropouts are experienced. Thus jitter is a key QoS impairment in real-time/multimedia applications and is a very useful metric to measure network and broadband performance [11]. In this work, jitter is one important performance metric that we use to assess the level of service continuity, disruption and glitches. We measure the average jitter experienced during handovers to evaluate and compare the performance of the proposed techniques.

2) Round-Trip Time: A network undergoing long round-trip times usually is the result of congestion, packet loss and queuing delays. Queued packets can cause variations in round-trip times and can not only increase delays but also packet loss rates. Larger round trip time (RTT) values indicate the average queue length at intermediate nodes is large. Lower RTT values imply less congestion and low average queue lengths. In this work, TCP RTT is used to assess service continuity, disruptions and glitches.

3) Queue Size: Queue size is another factor that affects service continuity during a handover. The more packets are queued at the bottleneck, the longer the round-trip times. It is essential to keep queue size low immediately following a handover to reduce RTTs and in turn jitter. In this work, queue size is evaluated to quantify service continuity, disruptions and glitches as well.

4) Throughput: Bandwidth constantly fluctuates and can vary depending on the transmission protocol used (i.e. TCP and UDP). If congestion is detected TCP backs off (slows down) and thus influences network bandwidth causing transmission rate fluctuations [12], [13]. Throughput is also a key metric essential for assessing network performance. In this work, throughput is evaluated to quantify service continuity, disruptions and glitches.

B. Test Topology and Result Analysis

In order to evaluate TCP-NR and TCP-CLAH, the aforementioned issues are analyzed to characterize network performance and service continuity during a handoff. Of

particular interest is the minimization of visible connectivity glitches due to handoff. Glitches involve a sudden increase or decrease in round-trip time and queue size which can be characterized by sudden fluctuations in jitter. A sudden change of the point of access network can cause significant glitches in a connection. Thus, trying to minimize these glitches and maintain a smooth transition is what we evaluate and resolve.

To perform our tests the topology in Figure 1 is configured in NS3. As illustrated, two networks of varying data rates are created with a common destination node. This can be viewed as a 54 Mbps Wifi network and a 5.5 Mbps cellular network. The maximum achievable throughput in our test topology is 19 Mbps for Network 1 and 3 Mbps for Network 2. This is expected since experimental speeds tend to be much less than theoretical link speeds [14].

The topology is configured in a way such that two paths are available from node ***Mobile*** (source) to node ***Dest*** (destination); that is one path per network. This helps emphasize the influence of handovers on a connection's performance and service continuity. Our tests have been encouraging and shine a light on the importance of cross-layer TCP adaptability.

Our tests focus on the aforementioned factors that influence connection glitches and service continuity during handovers. The conducted tests involve a mobile device moving from one network to another, starting in Network 1, every 15 seconds over a 100 second time period. Recall from Section IV we assume a soft-handover where cross-layer notifications inform TCP of a significant change in bandwidth. During this time period the mobile device has established a connection with the destination node and is transferring data (i.e. YouTube, P2P data transfers). The tests are conducted with varying application sending rates; 10, 15 and 20 Mbps.

Figures 2, 3 and 4 compare the jitter experienced by TCP-NR and TCP-CLAH over the connection for each respective sending rate. As illustrated, TCP-CLAH drastically improves jitter and hence disruptions and glitches are reduced during a handoff. TCP-CLAH improves jitter by roughly 3 times for an application send rate of 10 Mbps, 5 times for a 15 Mbps send rate, and about 75% for a 20 Mbps send rate. Disabling dupacks when a handoff occurs prevents unnecessary retransmissions which minimizes the variation in packet delay. In addition, resetting the RTT and RTO estimations allows for TCP to relearn the link conditions gracefully as shown in Figure 5. Finally, by setting the congestion window to the path BDP when a handoff occurs, we prevent rapid queuing at the bottleneck as mentioned in Section IV and illustrated in Figure 6 which in turn allows for shorter initial round-trip times. All these factors influence jitter which is an essential metric for characterizing and visualizing handoff induced glitches.

As we can see from our tests TCP-NR resulted in erratic behavior during network handovers. Specifically large fluctuations in jitter, RTT and queue size. This glitch although small can be visible by the user and can disrupt an otherwise problem free connection. By preemptively

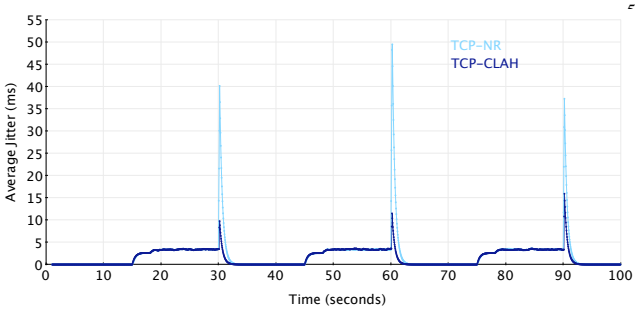

Fig. 2: Average jitter for 10 Mbps send rate

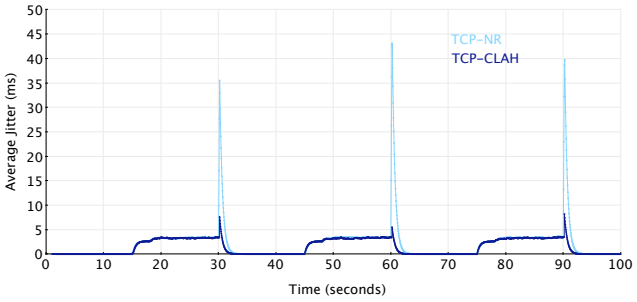

Fig. 3: Average jitter for 15 Mbps send rate

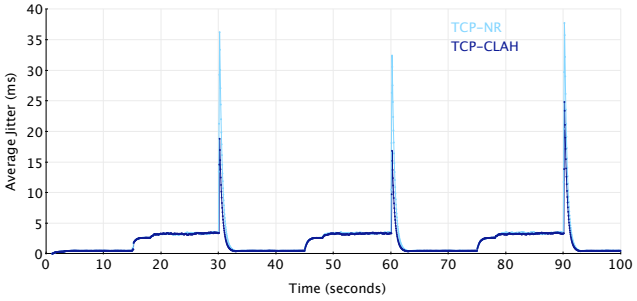

Fig. 4: Average jitter for 20 Mbps send rate

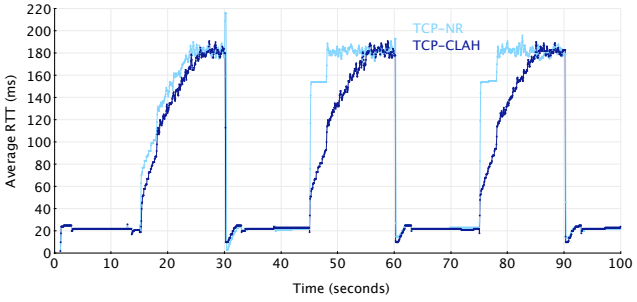

Fig. 5: Average RTT for 20 Mbps send rate

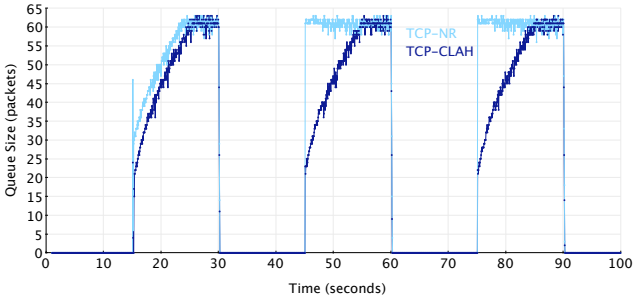

Fig. 6: Queue size of bottleneck interface for 20 Mbps send rate

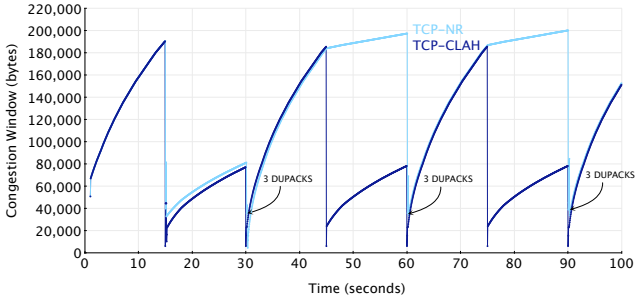

Fig. 7: Sender's congestion window for 20 Mbps send rate

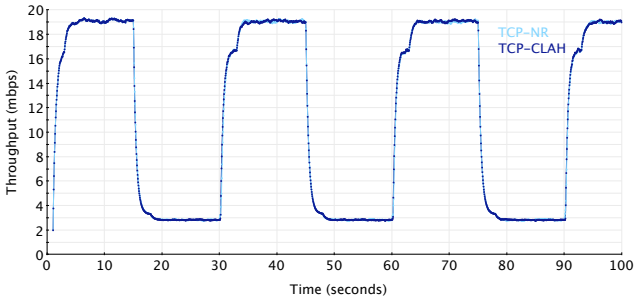

Fig. 8: Average throughput for 20 Mbps send rate

adjusting the congestion window to the path BDP when a handoff occurs we not only avoid rapid fluctuations in jitter but also drastically minimize connection disruptions and glitches without compromising throughput. Upon a handoff, the congestion window is much larger than the path BDP as shown in Figure 7. This is expected as the path BDP refers to the maximum number of bytes that can be transmitted without being queued at the bottleneck.

So far, we have showed that the proposed TCP-CLAH outperforms the existing TCP NewReno in terms of jitter, RTT, and queue size performances, thereby reducing disruptions and glitches of ongoing communications significantly during handoff. Now we show that TCP-CLAH achieves such performances without compromising the overall achieved throughput. To show this, we plot in Figure 8 the throughput achieved under both TCP-NR and TCP-CLAH while conducting these tests. The figure indeed shows the throughput is not compromised, as both TCPs achieve similar throughput performances.

VI. Conclusion

In this work we analyze TCP's drawbacks as a result of handovers between networks with varying data rates. A list of issues are categorized and presented. A modification to TCP NewReno is proposed based on the bandwidth delay product of the underlying networks. Our proposed cross-layer assisted handoff TCP (TCP-CLAH) is compared to TCP NewReno. Tests were performed to analyze and evaluate service continuity and emphasize the importance of incorporating the bandwidth-delay product within TCP. Our simulations show that TCP-CLAH outperforms TCP-NR significantly in terms of jitter, round trip time, and queue size, thereby improving communication quality (reducing disruptions and glitches) during handoff drastically.

VII. Acknowledgment

This work was supported in part by National Science Foundation under NSF award CNS-1162296.

References

[1] Carsten Burmeister, Ulrich Killat, and Jens Bachmann, "Tcp throughput optimized handover decisions," 2005.

[2] Tom Goff, James Moronski, D. S. Phatak, and Vipul Gupta, "Freeze-tcp: A true end-to-end tcp enhancement mechanism for mobile environments," in *In Proceedings of IEEE INFOCOM'2000, Tel Aviv*, 2000, pp. 1537–1545.

[3] L. Daniel and M. Kojo, "Employing cross-layer assisted tcp algorithms to improve performance with vertical handoffs," in *International Journal of Communications Networks and Distributed Systems,*, 2008.

[4] L. Daniel, I. Jarvinen, and M. Kojo, "Combating packet reordering in vertical handoff using cross-layer notifications to tcp," in *Networking and Communications, 2008. WIMOB '08. IEEE International Conference on Wireless and Mobile Computing,*, 2008.

[5] Xiaohuan Yan, Y. Ahmet ekerciolu, and Sathya Narayanan, "A survey of vertical handover decision algorithms in fourth generation heterogeneous wireless networks," *Computer Networks*, vol. 54, 2010.

[6] Kostas Pentikousis, "Tcp in wired-cum-wireless environments," *Fourth Quarter*, vol. 3, pp. 2–14, 2000.

[7] W. Hansmann and M. Frank, "On things to happen during a tcp handover," in *Local Computer Networks, 2003. LCN '03. Proceedings. 28th Annual IEEE International Conference on*, 2003.

[8] Kai Chen, Yuan Xue, Samarth H. Shah, and Klara Nahrstedt, "Understanding bandwidth-delay product in mobile ad hoc networks," *Comput. Commun.*, 2004.

[9] Douglas E. Comer, *Computer Networks and Internets*, Prentice Hall, 2008.

[10] Cisco Systems, "Understanding jitter in packet voice networks," `http://www.cisco.com/application/pdf/paws/18902/jitter_packet_voice.pdf`, February 2006.

[11] Villarreal S. Houmayoun F. Basu K. Munoz, D., "Heavy tail jitter in mobile packet networks," 2001, VTC '01.

[12] Michael J. Karels Van Jacobson, "Congestion avoidance and control," 1988, Proceedings of the Sigcomm '88 Symposium.

[13] E. Blanton M. Allman, V. Paxson, *http://www.ietf.org/rfc/rfc5681.txt*, 2009.

[14] Jinwoo Suh Mikyung Kang, Dong-In Kang, "Real-time support on 802.11 wiereless ad-hoc networks: Reality vs. theory," 2009, IEICE Trans. Commun.