Proactive Multi-Path TCP for Seamless Handoff in Heterogeneous Wireless Access Networks

Hassan Sinky*, Bechir Hamdaoui*, and Mohsen Guizani†
*Oregon State University, Email: sinkyh,hamdaoub@onid.orst.edu
† Qatar University, Email: mguizani@ieee.org

***Abstract*—Multi-Path TCP (MPTCP) is a new evolution of TCP that enables a single MPTCP connection to use multiple TCP subflows transparently to applications. Each subflow runs independently allowing the connection to be maintained if endpoints change; essential in a dynamic network. Differentiating between congestion delay and delay due to handoffs is an important distinction overlooked by transport layer protocols. Protocol modifications are needed to alleviate handoff induced issues in a growing mobile culture. In this article, findings are presented on transport layer handoff issues in currently deployed networks. MPTCP as a potential solution to addressing handoff- and mobility-related service continuity issues is discussed. Finally, a handoff-aware cross-layer assisted MPTCP (CLA-MPTCP) congestion control algorithm is designed and evaluated.**

I. Introduction

The demand for access to information has grown exponentially throughout the years. This accelerated growth has researchers focusing on integrating various wireless access technologies to provide higher data rates, more services and a global roaming culture. Naturally, in heterogeneous wireless networks (HetNets), mobile terminals are bound to undergo multiple handoffs and are equipped with multiple network interfaces to utilize the technologies they encounter. Handoffs can occur horizontally between networks of the same technology using the same interface or vertically between networks of different technologies using multiple interfaces. Due to the improvements in technology and overall power of mobile devices users are put in more mobile situations. As users become more mobile, drawbacks with TCP and other traditional protocols begin to arise. Disruptions in service continuity due to handoffs is an unacceptable outcome. Thus, it is essential to address these issues in mobile environments. Although technology is constantly evolving, TCP has remained mostly the same for more than 20 years [1]. In fact, 95% of all internet traffic is TCP [2]. These reasons have motivated the introduction of Multi-Path TCP (MPTCP) as a possible solution to the mobile scenario.

This work sheds light on overlooked handoff issues and proposes a Cross-Layer Assisted MPTCP (CLA-MPTCP) to alleviate resulting performance issues and glitches. Leveraging cross-layer information, where layers exchange data, improves MPTCP responsiveness, throughput and latency during fluctuating network conditions. Contributions of this work are:

- Performance evaluation and analysis of transport protocols during handoffs in HetNets.
- Establishing that reducing congestion windows at the time of handoff and utilizing cross-layer assistance is essential for ensuring service continuity.
- Design of handoff-aware cross-layer assisted MPTCP coupled congestion control algorithm.
- To our knowledge, this work is the first to investigate handoff connection disruptions and glitches with MPTCP, and to improve service continuity during handoffs through proactive cross-layer assisted SNR and BDP-based congestion window adjustments.

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

The rest of this paper is organized as follows. In Section II an experiment is conducted and findings are presented to analyze handoff issues in currently deployed wireless networks. MPTCP as a potential solution to handoff issues is discussed in Section III. CLA-MPTCP is presented and evaluated in Sections IV and V. Finally the article is concluded in Section VI.

II. Handoff Induced Issues in Current HetNets

Some work has been done in investigating transport layer issues during handovers such as those in [3]–[6]. For instance, traditional TCP has many handoff issues in terms of mobile data transfers, connection glitches and service continuity [7]. Sinky and Hamdaoui [7] examine and propose solutions to these issues using the New Reno variant of TCP which is the most widely used variant today. The challenge is to seamlessly maintain a mobile user's connection with little to no perceivable interruption in the event of a handoff. These issues are among many that have motivated the introduction of MPTCP. Since MPTCP utilizes multiple TCP subflows it is essential to understand basic TCP handoff induced issues.

Different TCP problems are amplified depending on the type of handoff that occurs [3]–[7]:

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

- **Packet re-ordering:** This issue occurs when the source node switches to a 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 acknowledgments (dupacks) to be sent. TCP would misinterpret 3 dupacks as being caused by congestion or loss and re-transmit packets incorrectly assumed to be lost when in reality this was caused by a handoff.
- **Inflated re-transmission timeout (RTO):** Naturally, a high-delay link will have high RTT and RTO values. Since the RTO is updated once every 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 longer. In other words, in the event of lost packets on a low-delay link, TCP would not recover quickly due to the high RTO value of the high-delay link. Thus, TCP must be able to recover lost packets faster using a smaller RTO representing the new low-delay link.

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

- **Spurious or false re-transmission timeout (RTO):** This issue arises when a handoff occurs 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 link 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:** In high-delay networks TCP requires several RTTs to probe available bandwidth. Specifically, after a handoff to a high-delay link the sender may inject more data onto the link than can be handled resulting in dropped packets. In addition, when TCP is in the slow start phase the congestion window grows exponentially. If a handoff occurs during the slow start phase a congestion window increase may result in packet losses and timeouts. This is referred to as a slow start overshoot. TCP may interpret this incorrectly and enter its congestion avoidance phase with a very small congestion window resulting in suboptimal TCP performance.

Transport layer protocols are neutral to the varying technologies of HetNets; rather, they react to the qualities of service and data rates available. Current deployed networks struggle to maintain service continuity, consistency and performance in mobile scenarios resulting in a frustrating user experience which is illustrated in II-A.

A. Transport Layer Analysis

A preliminary experiment was designed to force a mobile device into multiple handoffs. The HetNet in the EECS department at Oregon State University consists of 41 access points using both the 2.4 and 5 GHz bands and 802.11 a, b, g and n access technologies. The experiment was limited to the third floor, which comprised of 12 access points. A python script logged network measurements while the mobile device downloaded a large file and followed a path involving multiple access points as shown in Figure 1a. Figure 1 uses one of the experimental runs to illustrate these findings.

Access points throttle down transmit rates while mobile devices increase their power in an effort to maintain the connection resulting in performance degradation. Mobile terminals tend to cling to their current access point unless a handoff is necessary skipping intermediate access points across their path. This causes performance issues when multiple handoffs take place as shown by the erratic signal-to-noise ratio (SNR), throughput and RTT behavior around the points of handoff in Figure 1. Each vertical line indicates a channel switch or handoff logged by the mobile terminal as it moves across the path.

A mobile user can undergo multiple handoffs during an active connection's lifetime. This involves experiencing the quality degradation of a depleting data connection until a handoff is complete. That is, SNR and throughput decrease while RTTs increase as a handoff is approached and initiated. This is followed by a period of satisfactory connection quality until the connection begins to deplete again usually ending when a mobile user becomes stationary. This cycle greatly affects performance resulting in throughput gaps and service continuity issues for mobile users.

Transport layer protocols **must** be able to differentiate between delay caused by actual packet loss and congestion on a link and delay simply caused by handoffs. A combination of metrics must be considered for an accurate representation of network conditions. Section III briefly introduces MPTCP and some of its drawbacks.

III. Multi-Path TCP (MPTCP)

Despite being a relatively new protocol, MPTCP is becoming increasingly popular. With the release of iOS 7 in 2013, Apple became the first to implement MPTCP commercially. The novelty behind MPTCP is its ability to decouple TCP and IP to simultaneously use multiple TCP subflows and interfaces transparent to the application [1].

Among the main modes of operation available to MPTCP are `fullmptcp`, `backup` and `singlepath` [8], [9]. In `fullmptcp` mode the connection's data is striped among all subflows as space in the subflow windows becomes available where most of the data is sent on the least congested subflow (Linux MPTCP implements a path manager that schedules data on subflows with the lowest RTT) [9]–[11]. In `backup` mode slave subflows that have joined the current MPTCP connection are only used when the master subflow, used for initiating the MTPCP connection, fails [8], [12]. In `singlepath` mode a new subflow is established only when the interface goes down.

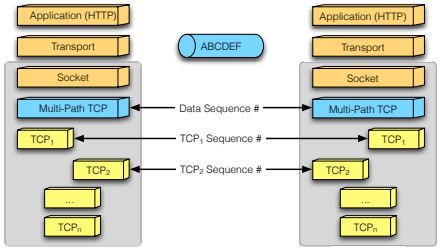


Figure 2: MPTCP architecture

A. MPTCP Architecture

A simplified overview of MPTCP can be seen in Figure 2. MPTCP's inherent architecture provides a potential solution to the aforementioned handoff issues. For instance, an application on a smartphone may use a single MPTCP connection to utilize both Wi-Fi and cellular interfaces to communicate with a server even if endpoints were to change, whereas traditional TCP would break.

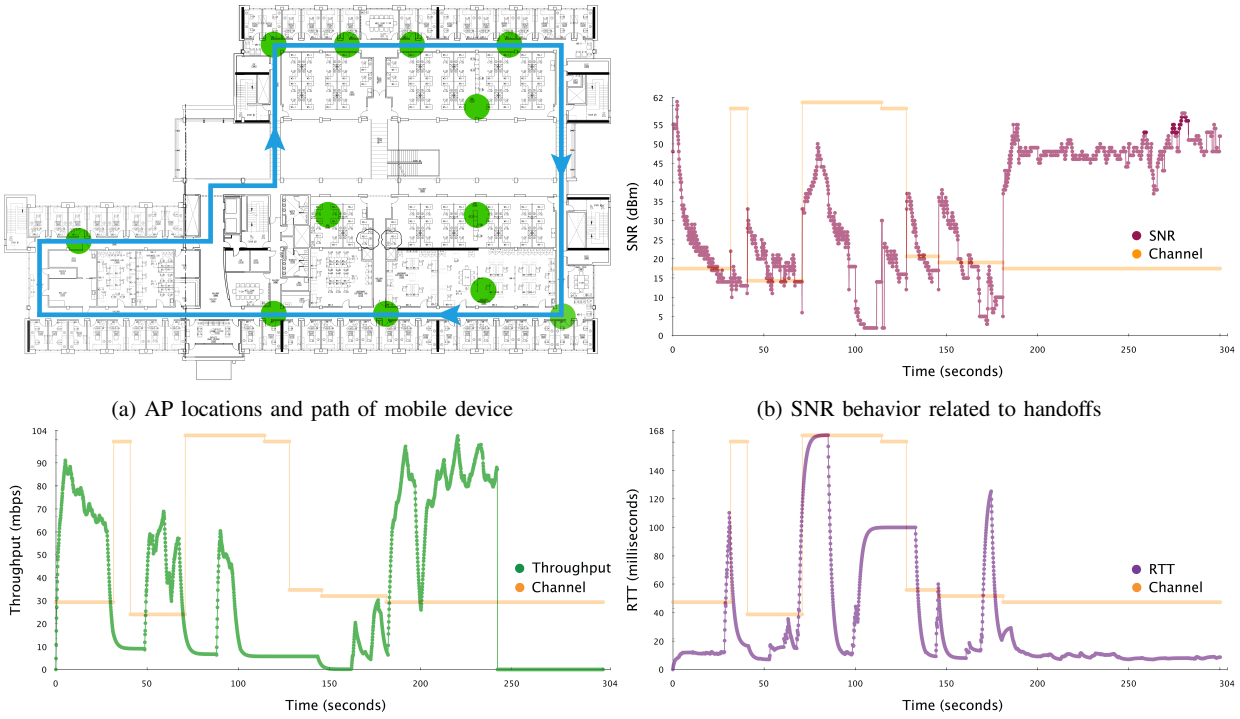
MPTCP subflows may appear or disappear at any time during an active connection. Similar to TCP, each subflow is initiated using a three-way handshake and operates independently of others with their own congestion states (i.e. cwnd) [12]. For reliable, in-order data delivery MPTCP uses a data sequence number (DSN) to number all data sent over a MPTCP connection and a per subflow sequence number (SSN) mapped to the DSN. This allows for the same data (DSN) to be re-transmitted on different subflows in the event of packet loss or failure. The mapping, once declared, is fixed and carried with the SSN sent. DSN are acknowledged using a data connection level acknowledgment.

MPTCP aims to simultaneously pool available resources, appearing as a single resource to the end user application, whilst greatly improving user experience. This is realized through three MPTCP goals listed in [13]:

- **Goal 1** (Improve throughput): Perform at least as well as single-path TCP would on its best subflow.
- **Goal 2** (Do no harm): Do not use more capacity than if it was single-path TCP on its best subflow.
- **Goal 3** (Balance congestion): MPTCP should move as much traffic as possible off its most congested paths.

B. MPTCP Congestion Control

Simply running standard TCP congestion control on each subflow gives MPTCP an unfair share of capacity when its subflows share a bottleneck [13]. By default, MPTCP implements a coupled congestion control algorithm that links subflow increase functions. For each ACK on subflow i, congestion window w_i is increased by $min(\frac{\alpha}{w_{total}}, \frac{1}{w_i})$. For each loss $w_i = \frac{w_i}{2}$. The α parameter controls the aggressiveness of a MPTCP connection and is chosen so that the MPTCP aggregate flow is equal to the achievable throughput of a single-path TCP flow on the best path [13]:



(a) AP locations and path of mobile device

(b) SNR behavior related to handoffs

(c) Throughput performance related to handoffs

(d) RTT behavior related to handoffs

Figure 1: Transport layer performance measurements related to multiple handoffs

$$\alpha = w_{total} \times \frac{\max\left(w_i/RTT_i^2\right)}{\left(\sum w_i/RTT_i\right)^2}$$

C. Drawbacks of MPTCP Congestion Control

Handoff issues presented in Section II still apply to the TCP subflows of MPTCP. For example, MPTCP reordering occurs both at the subflow and data levels which can cause added complexity with duplicate ACKs and spurious RTOs. Common MPTCP drawbacks that may arise from handoffs are listed below:

- **Responsiveness**: MPTCP relies on α to adapt to network conditions. However, to reduce computational costs, α is only updated when there is a packet drop or once per RTT [14]. In addition, sender RTT estimations can be inaccurate due to TCP's delayed ACK mechanism and smoothed RTT values (SRTT) especially when a network is over-buffered. This results in slow responsiveness to changes in subflow congestion windows resulting in an underestimation of α affecting MPTCP's increase rate. Thus, a depleting connection due to handoff may be misinterpreted by MPTCP as being highly congested. The time needed for RTT updates to characterize a congested link is detrimental for a highly mobile user. With the advantage of cross-layer assistance this issue can be minimized by taking into consideration physical network conditions such as signal strength and user mobility that otherwise are not available to MPTCP. By reducing the dependence on RTT and packet loss as a means to adjust MPTCP aggressiveness, sensitivity to a depleting connection due to handoffs is increased allowing for quicker adjustments to be made and more seamless service continuity.
- **Fullmesh gaps**: In fullmesh mode data that was originally striped onto a particular subflow is delayed or lost at the time of handoff and would need to be re-mapped onto other subflows. The worst case scenario is a subflow's entire congestion window is lost resulting in a gap in the received data sequence numbers. These data sequence numbers are then re-mapped to other available subflows causing glitches and affecting service continuity at the point of handoff. Delay is increased even more if RTOs are high forcing subflows to be momentarily inactive. Incorporating cross-layer assistance will allow for data originally mapped onto the conflicting subflow to be re-mapped earlier onto other available subflows during the scheduling process avoiding increased delays and large gaps in the received sequence numbers.
- **Backup delay**: In backup mode packets are transmitted on different subflows only if the master subflow fails. This is not ideal for handoff scenarios as it negatively affects service continuity by waiting for a connection to fail before using another [8]. The ability to detect a failing subflow through cross-layer assistance can allow MPTCP to utilize subsequent subflows prior to the master subflow failing.

Packet loss and RTT alone are not enough for a MPTCP connection to quickly adapt to handoffs. Thus, balancing between network congestion and impending handoff as a congestion control mechanism is necessary. A solution is required that incorporates not only real-time network monitoring but also cross-layer assistance in MPTCP. In the following section a Cross-Layer Assisted MPTCP (CLA-MPTCP) is proposed to alleviate handoff induced glitches and disruptions in service continuity.

IV. Cross-Layer Assisted Coupled Congestion Control

MPTCP requires the ability to differentiate between fluctuations in network conditions caused by congestion and those caused by mobility and network handoffs. Figure 3 shows subflow SNR of a typical MPTCP connection as a mobile host moves across a path

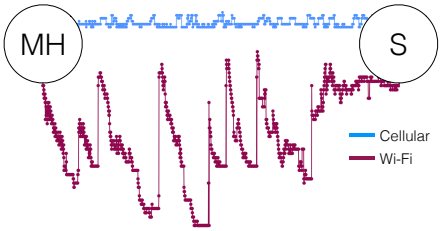


Figure 3: Subflow SNR behavior between mobile and server

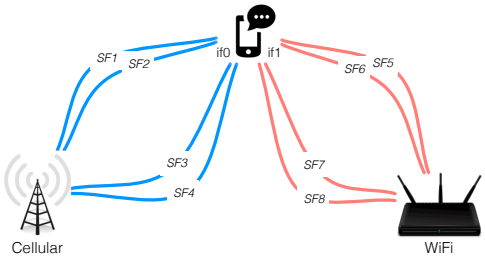


Figure 4: Subflows must go through same point of access

while communicating with a server. This results in a scenario where stable conditions are experienced on the cellular interface and erratic behavior on the Wi-Fi interface where gaps in SNR, throughput and service continuity are seen. As mentioned before, relying on RTT alone as a measure of link quality does not allow MPTCP to adapt quickly enough especially when RTT and RTO values are high.

In order to minimize connection glitches and maintain adequate service continuity throughout the handoff process the bandwidth delay product (BDP) must be considered [7]. 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 [15]. The BDP is represented by the product of a link's capacity and its RTT. By definition then the path BDP, BDP_{path}, is the product of the bottleneck throughput of the forward and backward paths, 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 and S' are the packet sizes in the forward and backward paths respectively. As done in [15], since $\sum_{i=1}^{n}\frac{S}{R_i} \leq \sum_{i=1}^{n}\frac{S}{R_{min}}$ and $\sum_{i=1}^{m}\frac{S'}{R'_i} \leq \sum_{i=1}^{m}\frac{S}{R_{min}}$, the path BDP can be loosely upper bounded by

$$BDP_{path} \leq R_{min}\left(\sum_{i=1}^{n}\frac{S}{R_{min}} + \sum_{i=1}^{m}\frac{S}{R_{min}}\right) = nS+mS = S\times N \quad (2)$$

where $N = n + m$. Note that having the path BDP exceed $S \times N$ results in packets being queued at the bottleneck [15]. In what follows, let $BDP_{UB} = S\times N$. At the time of handoff, where service continuity is crucial for a user's quality of experience (QoE), sending no more than the BDP_{UB} helps reduce queuing times, RTT values and jitter while maintaining adequate throughput [7]. In contrast to our previous work, which considered traditional single-path TCP, we shift our focus towards MPTCP and improving service continuity through proactive BDP and SNR-based subflow congestion window adjustments as a solution to the mobile handoff problem.

A. Design

Modifications to MPTCP are proposed to alleviate the issues presented in Section II. An additional MPTCP design goal is introduced:

- **Design goal** (Cross-layer assistance) A multi-path flow should consider both a subflow's signal strength and BDP to anticipate and alleviate handoff induced network conditions to maintain and improve service continuity.

In designing CLA-MPTCP three assumptions are made:

- A mobile device is equipped with two interfaces (i.e. cellular and Wi-Fi) where the master subflow is initiated on the Wi-Fi interface.
- Multiple handoffs are experienced on the Wi-Fi interface across a path.
- Both endpoints of a CLA-MPTCP connection exchange, through signaling, cross-layer information such as current and candidate access point signal strengths and handoff notifications.

CLA-MPTCP brings three modifications to MPTCP:

- Utilization of a signal strength-based subflow throughput model to prepare for handoffs while maintaining a connection-level throughput threshold R_0.
- Subflow congestion windows are adjusted to BDP_{UB} and RTT and RTO estimations are reset on handoff notifications.
- Couples interface signal strength and subflow throughput with MPTCP's congestion control algorithm to preemptively adjust the aggressiveness of individual subflows in anticipation of handoffs.

All subflows on a particular interface must go through their point of access network as shown in Figure 4. That is, a single interface cannot be connected to multiple cellular towers or multiple Wi-Fi access points. In the event of a handoff all subflows on the current interface must disconnect and reconnect on the next access network. For instance, cellular GSM networks, which make up more than 80 percent of all global mobile connections, implement hard handoffs where a connection must be broken until it is reestablished [16]. Conversely, handoffs between access points within the same Wi-Fi network can maintain their connection but still suffer from the consequences of a depleting connection. This requires more complex and adaptive subflow policies in order to maintain service continuity.

B. System Model

Compared to Wi-Fi, cellular connections experience higher latency and lower throughput while providing stability and less packet loss. On the other hand, the time it takes for a roaming client's Wi-Fi interface to scan and connect to available access points depends on many factors and can be detrimental to service continuity. IEEE standards such as $802.11k$ and r aim to streamline this process to help reduce roaming times. The $802.11k$ amendment reduces scan times by providing roaming clients with a neighbor report listing candidate access points. In addition, $802.11r$ expedites the authentication process by allowing a client to pre-authenticate with an access point potentially reducing authentication time from seconds to milliseconds.

Commercial implementations, such as those in Apple and Google

[1]RTT consists of propagation, transmission, queuing and processing delays only; i.e., end-to-end packet delay.

Parameter	Description
C_1	Capacity of Wi-Fi interface
C_2	Capacity of cellular interface
R_1^1	Current Wi-Fi throughput on AP_1
R_1^2	Current Wi-Fi throughput of candidate access point (AP_2)
R_1^h	Wi-Fi throughput at time of handoff
R_2	Current cellular throughput
R_2^p	Cellular throughput when pre-handoff is initiated
R_{tot}	Total connection throughput = $\sum R_i$
R_0	Connection throughput threshold
δ	$R_0 - R_1^1 - R_2$
S_1	Current signal strength of AP_1
S_2	Current signal strength of AP_2
Δ_0	Scan trigger threshold in dBm
Δ_1	Pre-handoff trigger threshold in dBm
Δ_2	Handoff trigger threshold in dBm

Table I: CLA-MPTCP system parameters

mobile devices, incorporate a signal strength trigger threshold[2] that, once reached, is used to initiate a scan for candidate access points provided by $802.11k$. Typically, a candidate access point with a signal strength differential of 12 dBm is selected triggering the handoff and pre-authentication using $802.11r$. A signal strength differential is necessary as to avoid a client from ping ponging between candidate access points. The proposed system model considers these aforementioned concepts in designing CLA-MPTCP.

The model parameters defined in Table I are integrated within CLA-MPTCP to adaptively maintain the connection threshold R_0 by conducting subflow signal strength-based throughput adjustments and provide a more seamless handoff transition. Using the measurement findings from Section II, CLA-MPTCP categorizes a mobile user into three distinct stages: stationary, pre-handoff and handoff. The design of these stages are discussed next.

1) Stationary: During the stationary stage CLA-MPTCP subflows are assumed to maintain stable throughput, that is, achieving their respective interface capacities C_i. In order to maintain connection threshold R_0, the CLA-MPTCP network module monitors and exchanges subflow goodput R_i and current and candidate AP signal strengths S_i between source and destination. Data is offloaded onto the cellular interface if $R_{tot} < R_0$ and $S_1 > S_2 + \Delta_1$ where cellular throughput $R_2 = \max(0, \min(\delta, C_2))$. The stationary process is illustrated in Figure 5. This differs from the pre-handoff stage where subflow rates are proactively adjusted based on their current rates and signal strengths.

2) Pre-Handoff: During this phase proactive preparations are made in anticipation of a handoff. This process assumes a depleting connection on the Wi-Fi interface where throughput decreases proportional to the signal strength. All internet routers contain buffers to hold packets in times of congestion [17] and due to immense bufferbloat in real networks [14], the amount of actual in-flight packets is much higher than the real path BDP. Thus, in order to reduce a subflow's throughput, in response to mobility and handoffs, the congestion window needs to be proactively reduced below the path BDP. The intuition behind this is to gradually reduce buffer sizes, queuing delays and round trip times in order to provide a more seamless transition.

CLA-MPTCP's subflow throughput, R_i, is modeled as a function of signal strength, S_i, using a system of linear equations. The design

[2]The trigger threshold is the minimum signal strength a client requires to maintain a connection. For example, Apple devices use an RSS trigger threshold of -70 dBm.

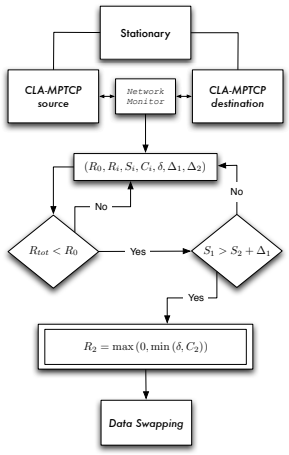


Figure 5: CLA-MPTCP stationary stage flow

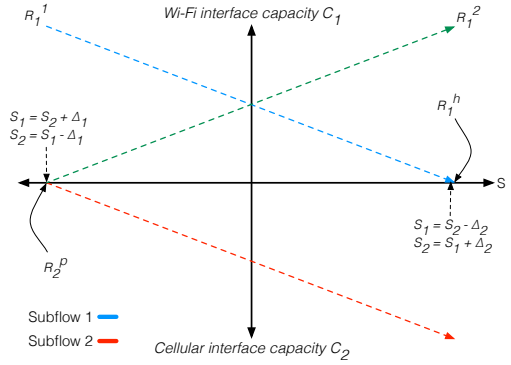


Figure 6: CLA-MPTCP system model

is formulated based on the desired behavior illustrated in Figure 6 where the x and y axes represent Wi-Fi signal strength over time and interface capacities respectively. When pre-handoff is initiated, that is, when conditions $R_{tot} < R_0$ and $S_1 < S_2 + \Delta_1$ are met, the Wi-Fi subflow is assumed to have maintained a stable throughput of C_1 up to this point. R_1^h is a tunable parameter representing the rate reached at the time of handoff. Note that prior to a handoff a Wi-Fi subflow still achieves usable throughput, where R_1^h may not necessarily be zero. During pre-handoff, the desired Wi-Fi subflow rate, R_1^1, can be derived using the following linear system:

$$R_1^1(S_1) = \begin{cases} C_1 = a \times (S_2 + \Delta_1) + b \\ R_1^h = a \times (S_2 - \Delta_2) + b \end{cases}$$

The adjusted pre-handoff rate for the Wi-Fi interface is then,

$$R_1^1 = \frac{C_1 - R_1^h}{\Delta_1 + \Delta_2} \times (S_1 - S_2 - \Delta_1) + C_1$$

In addition, upon entering pre-handoff, cellular throughput, R_2^p, is set to the current R_2 from the stationary phase. When a handoff is initiated, the amount of data offloaded onto the cellular interface is maximized to its respective capacity C_2 in order to reduce service continuity issues. As before, this gives the following linear system:

$$R_2(S_2) = \begin{cases} R_2^p = a \times (S_1 - \Delta_1) + b \\ C_2 = a \times (S_1 + \Delta_2) + b \end{cases}$$

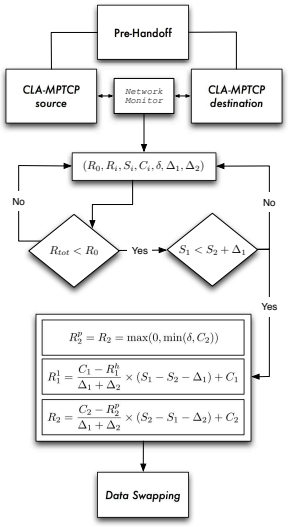


Figure 7: CLA-MPTCP pre-handoff stage flow

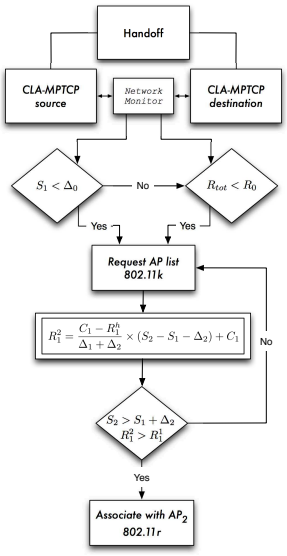


Figure 8: CLA-MPTCP handoff stage flow

The adjusted pre-handoff rate for the cellular interface is then,

$$R_2 = \frac{C_2 - R_2^p}{\Delta_1 + \Delta_2} \times (S_2 - S_1 - \Delta_2) + C_2$$

This phase provides CLA-MPTCP with the desired Wi-Fi and cellular subflow adjustments needed to prepare for and anticipate handoffs. CLA-MPTCP subflow congestion windows are then proactively adjusted using $BDP_{path} = R_{new} \times RTT_{path}$ where R_{new} is set to the newly derived throughputs, R_1^1 and R_2. This allows for R_0 to be maintained as long as possible until a handoff is initiated.

A large congestion window leading up to a handoff can be detrimental to service continuity [7]. Maintaining a large congestion window increases the probability of contention, RTT, queuing and packet loss which can degrade MPTCP performance drastically. By setting the congestion window to the adjusted path BDP, that is the maximum number of bytes that can be in transit across the path, guarantees the transmission of the maximum amount of bytes while avoiding additional packet queuing at intermediate links at the time of handoff. Figure 7 summarizes the pre-handoff process.

3) Handoff: When a handoff is initiated the Wi-Fi interface must disconnect and re-associate with a neighboring access point. Naturally, this causes the current throughput to momentarily be zero until a re-association completes. However, prior to the handoff the connection may still be achieving usable throughput. Traditionally, the scan and handoff trigger thresholds, Δ_0 and Δ_2, take precedence in deciding when to handoff. The handoff phase adds two checks to the traditional method. If Δ_0 has not been reached a request for an AP list is made if $R_{tot} < R_0$. Assuming the current and candidate access points share similar characteristics and capacities, the current rate of the candidate access point R_1^2 can similarly be modeled in terms of signal strength:

$$R_1^2(S_2) = \begin{cases} 0 = a \times (S_1 - \Delta_1) + b \\ C_1 = a \times (S_1 + \Delta_2) + b \end{cases}$$

Yielding,

$$R_1^2 = \frac{C_1 - R_1^h}{\Delta_1 + \Delta_2} \times (S_2 - S_1 - \Delta_2) + C_1$$

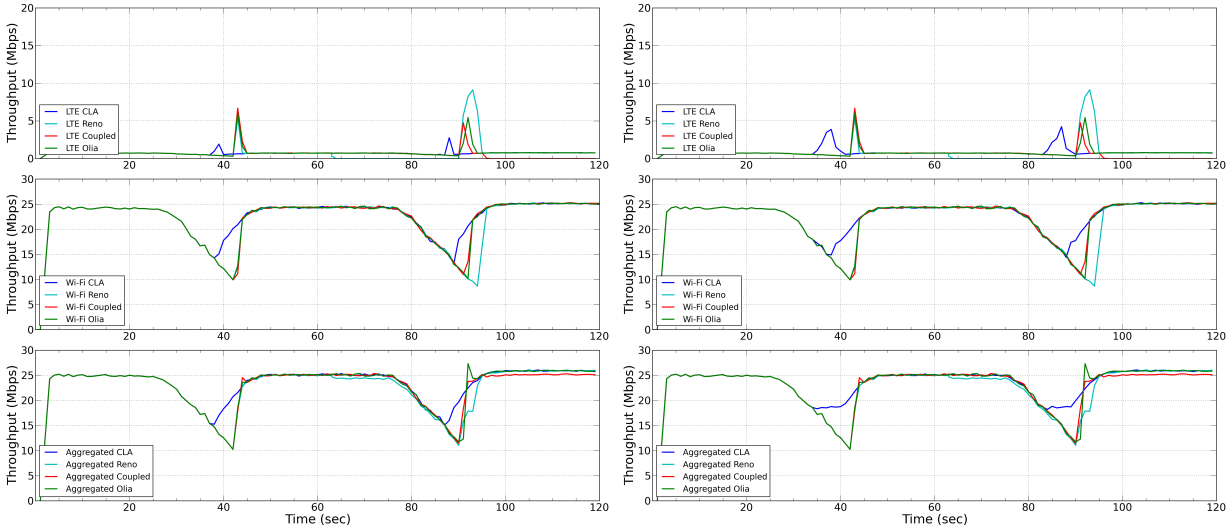
R_1^2 is continually monitored and compared to the current rate R_1^1. If $R_1^2 > R_1^1$ or $S_2 > S_1 + \Delta_2$, whichever is true first, an association with the candidate AP is initiated.

When a CLA-MPTCP sender receives a subflow handoff notification it resets RTT and RTO estimations. The intuition behind this is that if the RTO is set too high MPTCP may be slow to recover lost packets. If set too low, spurious RTOs cause MPTCP to assume packets are lost when in reality they may simply be on their way. Analyses established that by resetting these values MPTCP converges quicker to the new network's RTT and RTO estimations. Depending on the subflow TCP variant, duplicate acknowledgments are temporarily disabled to avoid false dupacks caused by packet reordering. Finally, the congestion window of the subflow is set to the respective path BDP to avoid added queuing at the bottleneck as well as link overshoots. Using the path BDP ensures throughput is not compromised as shown in Section V.

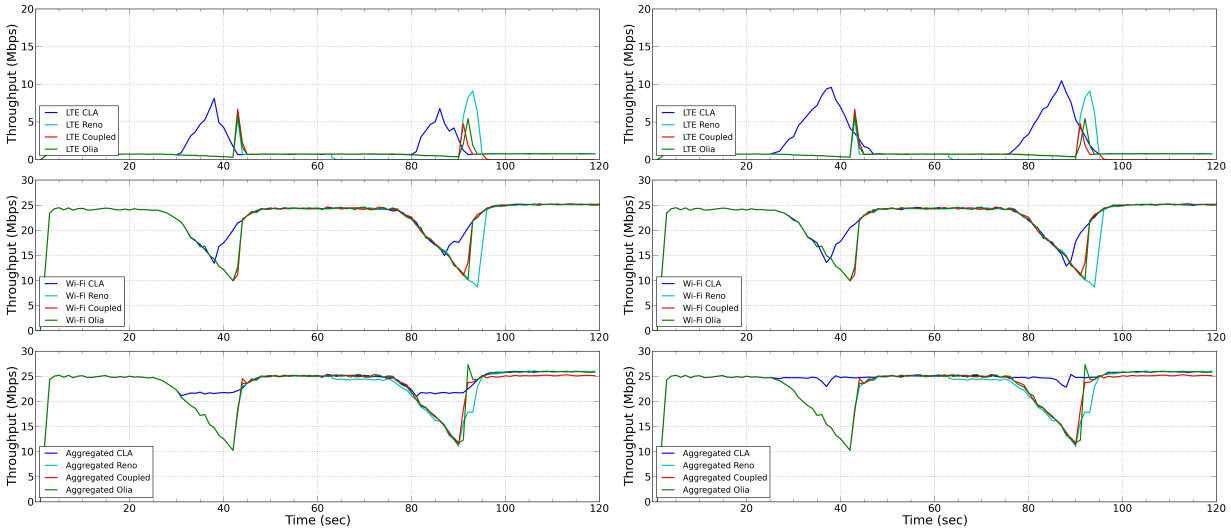
In Section V service continuity and performance measurements (i.e. aggregate throughput, RTT, jitter, re-transmissions and duplicate ACKs) are evaluated.

V. CLA-MPTCP Evaluation

To evaluate CLA-MPTCP the network topology in Figure 11a is built using NS3-DCE. DCE is a framework that allows for existing Linux `kernelspace` network protocols to be used within NS3 simulations [18]. In addition, an access point decision algorithm is implemented in NS3 that uses an SNR-based scan trigger threshold, Δ_0, of 20 dBm. Once reached, the mobile client scans for candidate access points and initiates an association if $S_2 > S_1 + \Delta_2$ where $\Delta_2 = 12$ dBm. CLA-MPTCP modifications were implemented in version 0.87 of the MPTCP Linux Kernel implementation and installed on NS3 nodes using DCE. Each CLA-MPTCP user is equipped with two interfaces, i.e. Wi-Fi and cellular. Similar to the analysis scenario in Section II-A, the mobile client moves across a path and undergoes two Wi-Fi handoffs while remaining connected to the LTE network. Data is sent from the mobile source to the destination at 30 Mbps where the capacity and delay of the Wi-Fi and LTE networks are set to 25 and 18 Mbps, 5 and 20 milliseconds respectively. In addition, R_1^h and Δ_1 are set to 5 Mbps and 25 dBm. The selection of the emulation parameters shown in Table II

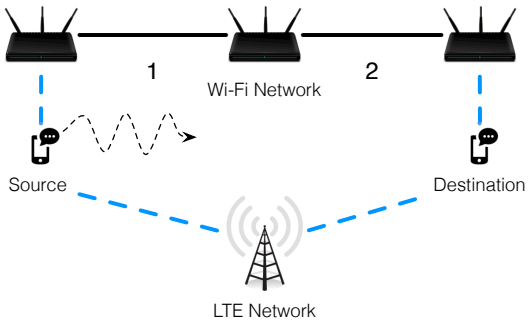


(a) $R_0 = 15$ (b) $R_0 = 18$

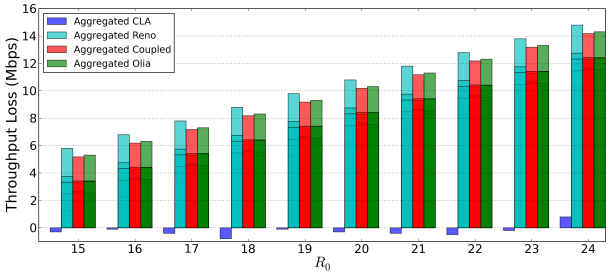


(c) $R_0 = 21$ (d) $R_0 = 24$

Figure 9: CLA-MPTCP throughput performance using different R_0 values compared to Uncoupled Reno, Coupled, and Olia using traditional handoff scheme with $\Delta_0 = 20$ and $\Delta_2 = 12$ dbm.



(a)



(b)

Figure 11: (a) NS3-DCE Network Topology used for simulations. (b) Throughput loss at handoff as a function of R_0.

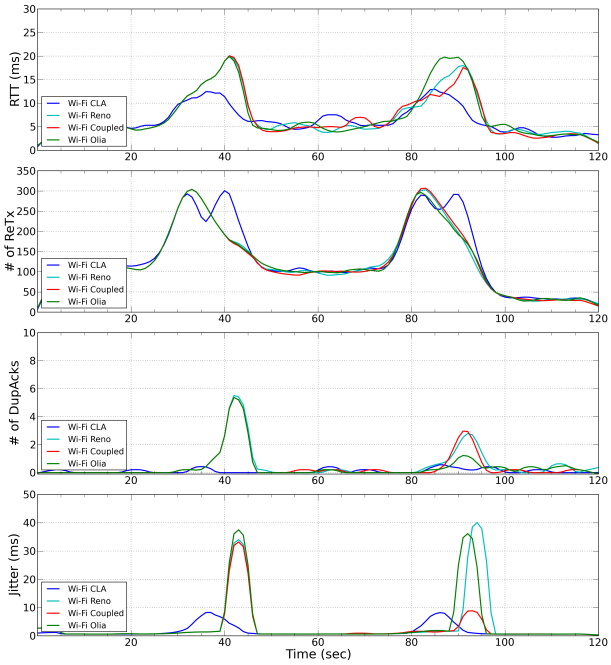


Figure 10: Average RTT, re-transmissions, duplicate ACKs and jitter measurements

were chosen based on our previous measurement study of currently deployed networks conducted in Section II and Figure 1b as well as manufacturer suggestions. For instance, parameters Δ_0 and Δ_2 were selected based on manufacturer recommendations of what is considered a suitable scan and handoff trigger threshold [19], [20]. Conversely, Δ_1 is an implementation specific decision chosen to balance the detection sensitivity between the stationary and pre-handoff phases. Thus, having $\Delta_1 > \Delta_0$ allows for CLA-MPTCP to enter the pre-handoff phase prior to scanning for candidate access points. Multiple simulations are run using different R_0 (connection throughput threshold) values ranging from 10 Mbps to C_1. Finally, network information such as current and candidate access point signal strength, subflow goodput and handoff notifications are exchanged between source and destination. During the CLA-MPTCP connection, the NS3-DCE mobile devices periodically log access point signal strengths in order to calculate the equations associated with the current phase. Algorithm 1 is applied in the stationary stage whereas algorithms 2 and 3 are applied in the pre-handoff and handoff stages. Once calculated the updated rates are exchanged with the CLA-MTPCP congestion control module for proactive congestion window adjustments. Information exchange can be implemented either using MPTCP's header option or through the use of Linux kernel external variables. For simplicity, the emulation environment is setup to utilize MPTCP specific external variables as well as toggling subflow binary variables for handoff notifications thus adding negligible overhead. Note MPTCP's second goal limits the aggregate throughput to no more than what is achievable by a single TCP flow on its best subflow (Wi-Fi).

The achieved aggregate goodput of CLA-MPTCP is compared with Uncoupled, Coupled and Olia (Opportunistic Linked Increases Algorithm) versions of MPTCP. Uncoupled MPTCP runs traditional TCP Reno on its individual subflows whereas coupled MPTCP implements the coupled congestion control algorithm which links the increase functions of MPTCP subflows introduced in Section III-B. Olia increases good quality subflows with small windows faster while

Algorithm 1 CLA-MPTCP Stationary Stage

Input: $R_0, S_i, C_i, \Delta_1, \Delta_2, \delta, R_{tot}$
while $S_1 > S_2 + \Delta_1$ and $R_{tot} < R_0$ **do**
  for each subflow i **do**
    if path index == cellular interface **then**
      $R_{tot} += R_i$
      let $\delta = R_0 - R_1^1 - R_2$
      let $R_2 = \max(0, \min(\delta, C_2))$
      let $R_{new} = R_2$
    end if
    if path index == Wi-Fi interface **then**
      $R_{tot} += R_i$
      let $R_{new} = R_1^1$
    end if
    calculate

$$BDP_{path} = \frac{R_{new} \times RTT_i}{MSS_i}[pkts]$$

    set $cwnd_i$ = BDP_{path}
  end for
end while
re-calculate α using adjusted cwnds

Algorithm 2 CLA-MPTCP Pre-Handoff Stage

Input: $R_0, S_i, C_i, R_1^h, R_2^p, \Delta_1, \Delta_2, R_{tot}$
From 802.11k AP report:
let S_1 = current AP_1 signal strength
let S_2 = candidate AP_2 signal strength
while $R_{tot} < R_0$ and $S_1 < S_2 + \Delta_1$ **do**
  for each subflow i **do**
    if path index == cellular interface **then**
      $R_{tot} += R_i$
      calculate

$$R_2 = \frac{C_2 - R_2^p}{\Delta_1 + \Delta_2} \times (S_2 - S_1 - \Delta_2) + C_2$$

      let $R_{new} = R_2$
    end if
    if path index == Wi-Fi interface **then**
      $R_{tot} += R_i$
      calculate

$$R_1^1 = \frac{C_1 - R_1^h}{\Delta_1 + \Delta_2} \times (S_1 - S_2 - \Delta_1) + C_1$$

      let $R_{new} = R_1^1$
    end if
    calculate

$$BDP_{path} = \frac{R_{new} \times RTT_i}{MSS_i}[pkts]$$

    set $cwnd_i$ = BDP_{path}
  end for
end while
re-calculate α using adjusted cwnds

Parameter	Value
C_1	25 Mbps
C_2	18 Mbps
R_1^h	5 Mbps
R_0	[10-24] Mbps
Δ_0	20 dBm
Δ_1	25 dBm
Δ_2	12 dBm

Table II: CLA-MPTCP emulation parameters

Algorithm 3 CLA-MPTCP Handoff Stage

Input: $R_0, R_1^h, R_1^1, S_i, C_i, \Delta_1, \Delta_2, R_{tot}$
while $S_1 < \Delta_0$ or $R_{tot} < R_0$ **do**
    request 802.11k AP report
    let S_2 = candidate AP_2 signal strength
    calculate

$$R_1^2 = \frac{C_1 - R_1^h}{\Delta_1 + \Delta_2} \times (S_2 - S_1 - \Delta_2) + C_1$$

    if $S_2 > S_1 + \Delta_2$ or $R_1^2 > R_1^1$ **then**
        initiate 802.11r authentication
        associate with AP_2
        notify CLA-MPTCP sender of handoff
    end if
end while
...
if Handoff notification on subflow i **then**
    reset RTT_i and RTO_i estimations
    temporarily disable dupacks
    obtain path BDP_{UB}
        set $cwnd_i$ = BDP_{UB}
end if

subflows with maximum windows increase slower. In other words, Olia re-forwards traffic from fully used paths to paths that have free capacity [21]. CLA-MPTCP integrates stationary, pre-handoff and handoff stages whereas existing implementations incorporate the traditional handoff method with $\Delta_0 = 20$ and $\Delta_2 = 12$ dBm.

Figure 9 shows LTE, Wi-Fi and aggregate throughput for different R_0 values. With different R_0 values the amount of data offloaded onto the LTE subflow as well as aggregate throughput can be tuned. That is, as R_0 increases more data is offloaded near handoffs resulting in a seamless transition for the end user. Results show CLA-MPTCP quickly adapts to network conditions and proactively offloads data from the Wi-Fi subflow to the LTE subflow. CLA-MPTCP's ability to maintain R_0 as well as preemptively tune subflow congestion windows to the adjusted path BDP ensures improved throughput and softens the transition. Essentially, R_0 is a parameter that fills the gaps in throughput caused by handoffs to improve a mobile device's service continuity.

Figure 11b shows the amount of throughput loss at the point of handoff as a function of R_0 using different Δ_2 values ranging from 1 to 15 dBm. CLA-MPTCP will try to maintain at least R_0 throughout the handoff process. If, at the time handoff, the aggregate throughput is greater than R_0 a negative throughput loss is experienced. For example, when $R_0 = 18$, CLA-MPTCP achieves an average aggregate throughput of 18.8 Mbps which results in a negative throughput loss or, in other words, a throughput improvement. Conversely, when $R_0 = 24$, the aggregate throughput cannot exceed the maximum achievable throughput at the time of handoff, $R_1^h + C_2$, which is less than R_0 resulting in a positive throughput loss. CLA-MPTCP's integration of a connection level threshold results in less throughput loss near handoffs where existing implementations lose throughput drastically as R_0 increases due to the lack of cross-layer assistance and handoff awareness.

Since current Linux MPTCP congestion control algorithms, uncoupled, coupled and Olia, rely on a path's RTT and packet losses when striping an input stream among the available subflows they do not react fast enough to sudden handoffs across a mobile user's path. The RTT of a subflow at the point of handoff may still be less than the RTT on other subflows causing MPTCP to still prefer and cling to the depleting subflow. In other words, the path scheduler within Linux MPTCP schedules data on the subflow with the lowest RTT value. For instance, although uncoupled Reno utilizes two separate subflows it still schedules most of the data on path with the lowest RTT value which happens to be the Wi-Fi network in the emulation environment. This results in similar behavior in their observed performances in Figures 9 and 10.

Figure 10 shows RTT, re-transmission, duplicate acknowledgment and jitter behavior of CLA-MPTCP. A near 50% and 75% improvement in RTT and jitter can be seen around the points of handoffs. In addition, CLA-MPTCP sees improvements in the number of packet re-transmissions and duplicate acknowledgments experienced. A sharp dip in packet re-transmissions can be seen at the time of handoff as well as stable and low occurrences of duplicate acknowledgments. Conversely, a sharp increase in packet re-transmissions occurs following a handoff due to the mobile device re-associating with a candidate AP when $R_1^2 > R_1^1$. This condition is met to prevent waiting for the trigger threshold, Δ_0, and candidate AP differential, Δ_2, to be reached. The re-association and utilization of the Wi-Fi network occurs earlier near the edge of the candidate AP's range which explains the temporary increase in packet re-transmissions following handoffs. The proactive congestion window adjustments made by CLA-MTPCP results in less packets being injected into the network near handoffs and a less strenuous transition.

VI. Conclusion

In this paper, issues dealing with handoffs in HetNets were discussed. Specifically weaknesses of transport layer protocols in mobile scenarios where multiple handoffs are imminent. A series of real world experiments in current deployed networks were conducted exploiting traditional TCP in mobile scenarios. MPTCP as a potential solution to the handoff issue is discussed. Possible drawbacks of MPTCP in increasingly mobile scenarios are presented. An additional MPTCP design goal is suggested emphasizing the need for cross-layer assistance. Finally, Cross-Layer Assisted MPTCP (CLA-MPTCP) is designed and evaluated. Results show the benefits of cross-layer assistance and integration of a connection level threshold and subflow signal strength as MPTCP outperforms uncoupled, coupled and olia congestion control algorithms in scenarios where multiple handoffs take place.

References

[1] O. Bonaventure, M. Handley, and C. Raiciu, "An overview of multipath tcp," in *USENIX*, October 2012.

[2] S. Barre, C. Paasch, and O. Bonaventure, "Multipath tcp: From theory to practice," in *IFIP Networking, Valencia*, May 2011.

[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] W. Hansmann and M. Frank, "On things to happen during a tcp handover," in *Proceedings of the 28th LCN Conference*. IEEE Computer Society, 2003.

[6] C. Burmeister, U. Killat, and J. Bachmann, "Tcp throughput optimized handover decisions," 2005.

[7] H. Sinky and B. Hamdaoui, "Cross-layer assisted tcp for seamless handoff in heterogeneous mobile wireless systems," in *IEEE GlobeCom*, Dec. 2013.

[8] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure, "Exploring mobile/wifi handover with multipath tcp," in *SIGCOMM '12 CellNet*. New York, NY, USA: ACM, 2012, pp. 31–36.

[9] *http://www.multipath-tcp.org/*, 2014.

[10] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How hard can it be? designing and implementing a deployable multipath tcp," in *Proceedings of the 9th USENIX Conference*, ser. NSDI'12, Berkeley, CA, USA, 2012.

[11] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental evaluation of multipath tcp schedulers," in *Proceedings of the 2014 ACM SIGCOMM Workshop on Capacity Sharing Workshop*, ser. CSWS '14. New York, NY, USA: ACM, 2014, pp. 27–32. [Online]. Available: http://doi.acm.org/10.1145/2630088.2631977

[12] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "Tcp extensions for multipath operation with multiple addresses," RFC 6824, Jan. 2013. [Online]. Available: https://tools.ietf.org/html/rfc6824

[13] C. Raiciu, M. Handley, and D. Wischik, "Coupled congestion control for multipath transport protocols," RFC 6356, Jan. 2011. [Online]. Available: https://tools.ietf.org/html/rfc6356

[14] Y.-C. Chen and D. Towsley, "On bufferbloat and delay analysis of multipath tcp in wireless networks," in *International Federation for Information Processing IFIP, Norway*, 2014.

[15] K. Chen, Y. Xue, S. H. Shah, and K. Nahrstedt, "Understanding bandwidth-delay product in mobile ad hoc networks," *Comput. Commun.*, 2004.

[16] M. Sauter, *Universal Mobile Telecommunications Systems (UMTS) and High-Speed Packet Access (HSPA)*. John Wiley and Sons, Ltd, 2011, pp. 115–204. [Online]. Available: http://dx.doi.org/10.1002/9780470978238.ch3

[17] G. Appenzeller, I. Keslassy, and N. McKeown, "Sizing router buffers," in *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '04. New York, NY, USA: ACM, 2004, pp. 281–292. [Online]. Available: http://doi.acm.org/10.1145/1015467.1015499

[18] D. Camara, H. Tazaki, E. Mancini, T. Turletti, W. Dabbous, and M. Lacage, "DCE: Test the real code of your protocols and applications over simulated networks," 2014.

[19] *http://support.apple.com/kb/ht5977*, 2014.

[20] I. Cisco Systems, "Enterprise best practices for apple devices on cisco wireless lan," Feb. 2016. [Online]. Available: http://www.cisco.com

[21] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec, "Mptcp is not pareto-optimal: Performance issues and a possible solution," vol. 21, no. 5, Oct 2013, pp. 1651–1665.

[22] Y. sup Lim, Y.-C. Chen, E. Nahum, D. Towsley, and K.-W. Lee, "Cross-layer path management in multi-path transport protocol for mobile devices," in *INFOCOM, 2014 Proceedings IEEE*, April 2014, pp. 1815–1823.

[23] X. Zhang, T.-M.-T. Nguyen, and G. Pujolle, "A cross-layer approach to optimize the performance of concurrent multipath transfer in wireless transmission," in *IFIP Conference*, ser. WD'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 107–111.

[24] S. F. Chien, H. Liu, A. L. Y. Low, C. Maciocco, and Y. L. Ho, "Smart Predictive Trigger for Effective Handover in Wireless Networks." *ICC*, pp. 2175–2181, 2008.

[25] W. Guan, X. Ling, X. S. Shen, and D. Zhao, "Handoff trigger table for integrated 3G/WLAN networks," in *IWCMC '06*, Jul. 2006.

[26] C. Raiciu, S. Barré, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving datacenter performance and robustness with multipath TCP," in *SIGCOMM '11*. ACM Request Permissions, Aug. 2011.

[27] "Ieee std 802.11k-2008 (amendment to ieee std 802.11-2007)," June 2008, pp. 1–244.

[28] H. Sinky and B. Hamdaoui, "Optimized link state routing for quality-of-service provisioning: implementation, measurement, and performance evaluation," vol. 14, no. 10, 2014, pp. 937–948. [Online]. Available: http://dx.doi.org/10.1002/wcm.2246

[29] H. Sinky, B. Hamdaoui, and M. Guizani, "Handoff-aware cross-layer assisted multi-path TCP for proactive congestion control in mobile heterogeneous wireless networks," in *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, 2015, pp. 1–7. [Online]. Available: http://dx.doi.org/10.1109/GLOCOM.2014.7417490

[30] H. Sinky and B. Hamdaoui, "Implementation and performance measurement and analysis of OLSR protocol," in *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, IWCMC 2010, Caen, France, June 28 - July 2, 2010*, 2010, pp. 286–290. [Online]. Available: http://doi.acm.org/10.1145/1815396.1815463

Hassan Sinky is a PhD candidate in the School of Electrial Engineering and Computer Science at Oregon State University. He received his Bachelors and Masters degrees in Computer Science at Oregon State University in 2007 and 2010, respectively. His current research interests include wireless mesh networks, the internet transport layer and mobility, and network handoffs in heterogeneous wireless networks.

Bechir Hamdaoui (S'02-M'05-SM'12) is presently an Associate Professor in the School of EECS at Oregon State University. He received the Diploma of Graduate Engineer (1997) from the National School of Engineers at Tunis, Tunisia. He also received M.S. degrees in both ECE (2002) and CS (2004), and the Ph.D. degree in Computer Engineering (2005) all from the University of Wisconsin-Madison. His current research focus is on distributed resource management and optimization, parallel computing, cooperative & cognitive networking, cloud computing, and Internet of Things. He has won the NSF CAREER Award (2009), and is presently an AE for IEEE Transactions on Wireless Communications (2013-present), and Wireless Communications and Computing Journal (2009-present). He also served as an AE for IEEE Transactions on Vehicular Technology (2009-2014) and for Journal of Computer Systems, Networks, and Communications (2007-2009). He served as the chair for the 2011 ACM MobiCom's SRC program, and as the program chair/co-chair of several IEEE symposia and workshops (including GC 2016, ICC 2014, IWCMC 2009-2016, CTS 2012, PERCOM 2009). He also served on technical program committees of many IEEE/ACM conferences, including INFOCOM, ICC, GLOBECOM, and others. He is an IEEE Communication Society (ComSoc) Distinguished Lecturer and a Senior Member of IEEE, IEEE Computer Society, IEEE Communications Society, and IEEE Vehicular Technology Society.

Mohsen Guizani (S'85-M'89-SM'99-F'09) is currently a Professor and the Chair of the Electrical and Computer Engineering Department at the University of Idaho, USA. Previously, he worked in Western Michigan University, University of West Florida, Kuwait University and Qatar University. He received his B.S. (with distinction) and M.S. degrees in Electrical Engineering; M.S. and Ph.D. degrees in Computer Engineering in 1984, 1986, 1987, and 1990, respectively, from Syracuse University, Syracuse, New York. His research interests include Computer Networks, Wireless Communications and Mobile Computing, Security and Internet of Things. He currently serves on the editorial boards of six technical Journals. He is the author of eight books and more than 400 publications in refereed journals and conferences. Dr. Guizani is an IEEE Fellow and a Senior member of ACM.