A Review of Multihoming Issues Using the Stream Control Transmission Protocol

T. Daniel Wallace and Abdallah Shami

***Abstract*—Known as multihoming, devices with more than one network interface can enhance their performance capabilities by harnessing unused resources from alternative access networks. Whether it's improved reliability or sheer throughput potential, network devices will benefit from a multihomed framework. Unfortunately, our current means of guaranteeing reliability while maintaining quality control, specifically, the transmission control protocol (TCP), does not support multihoming. Despite the latter, a relatively young transport layer standard called the stream control transmission protocol (SCTP), incorporates multihoming into its design. In this paper, we investigate state-of-the-art multihoming techniques using SCTP. A comprehensive survey of developments has brought forth three main research areas, namely: handover management, concurrent multipath transfer (CMT), and cross-layer activities. While the presented algorithms may offer sufficient results, many open problems still remain.**

***Index Terms*—SCTP, multihoming, concurrent multipath transfer, transport protocols.**

I. Introduction

IN THE PAST decade, advancements in wireless communications have reached unprecedented heights. The achievements made in wireless technology have provided pervasive network connectivity not only to the home and workplace, but also to remote areas where no wired infrastructure can reach. More recently, we have regarded this new means of communication as a primary resource and subscribe to it daily—if not constantly—in the form of mobile on-demand information services. Several wireless access technologies currently exist, such as: CDMA2000 and Wideband Code Division Multiple Access (W-CDMA) for 3G and 4G cellular communications; WiMAX (IEEE 802.16) for broadband access in metropolitan area networks (MANs); and Wi-Fi (IEEE 802.11) for wired equivalent local area networks (LANs). Nevertheless, the demand for higher data rates continues to grow, and researchers must find new ways to satisfy this need. One solution, known as multihoming, incorporates multiple network interfaces into a single device. Applied to wireless networking, multihoming can improve performance by exploiting unused resources from the radio spectrum.

Already many popular consumer electronics, like Apple's iPhone and Research in Motion's Blackberry Bold, come standard with Wi-Fi and cellular technologies (e.g., GSM and UMTS). Although products like these have more than one network interface, some advantages to multihoming are not being realized. Connection migration from one access technology to another, called vertical handover, is a perfect example of multihoming at its finest. Assuming a voice over IP (VoIP) telephone call is initiated within a Wi-Fi network, without a continuous series of overlapping Wi-Fi networks, the call will be dropped as soon as the phone travels even a short distance (e.g., 10s of metres). Although Wi-Fi offers high data rates at low cost, while mobile, cellular technologies can keep calls active; albeit at a higher cost with lower data rates. This is not to say these devices are not currently capable of such function, but at this time they do so only through proprietary means.

Unfortunately, our current means of guaranteeing reliability while maintaining quality control, specifically, the transmission control protocol (TCP), does not support multihoming capabilities. TCP, rather, binds a transport layer session to a single IP address; changing the IP address will kill any active session. Despite the latter, a relatively young transport layer standard called the stream control transmission protocol (SCTP), incorporates multihoming into its design. Developed by the Internet Engineering Task Force (IETF), SCTP already has a decade worth of research behind it.

It is the objective of this article to categorize and present the current research in SCTP as it pertains to multihomed devices with a strong interest in challenges as well as developments for the most salient issues; particularly, handover management, concurrent multipath transfer (CMT), and cross-layer activities. The rest of the paper is organized as follows. First, we present a review of the SCTP standard, followed by an introduction to multihoming and transport layer mobility. Next, we investigate the open problems facing SCTP under different multihomed scenarios. In addition, we survey and classify the recently proposed solutions to these problems. In conclusion, we comment on the state of this new research area while making some of our own predictions for its success.

II. Overview of the Stream Control Transmission Protocol

Developed by the IETF, SCTP [1] is a transport layer protocol that extends the functionality of the celebrated TCP standard. SCTP is a message oriented data delivery service providing full duplex connections and congestion control mechanisms. Similar to TCP, some of SCTP's features include but are not limited to: congestion and flow control, reliable data transfer, and ordered data delivery. SCTP, moreover, provides a hybrid service called partial reliable SCTP (PR-SCTP) [2], not to be confused with the user datagram protocol

Manuscript received 16 July 2010; revised 3 December 2010 and 2 April 2010.

The authors are with the Department of Electrical and Computer Engineering, The University of Western Ontario, London, Ontario, Canada (e-mail: ashami@eng.uwo.ca).

Digital Object Identifier 10.1109/SURV.2011.051111.00096

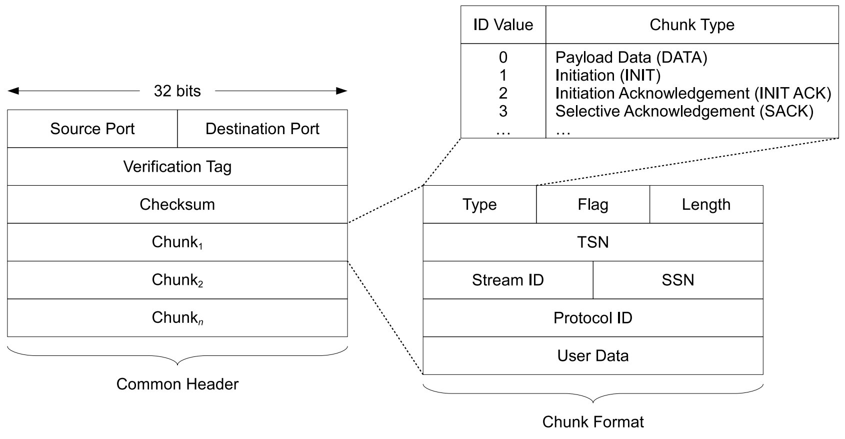

Fig. 1. SCTP's segment format.

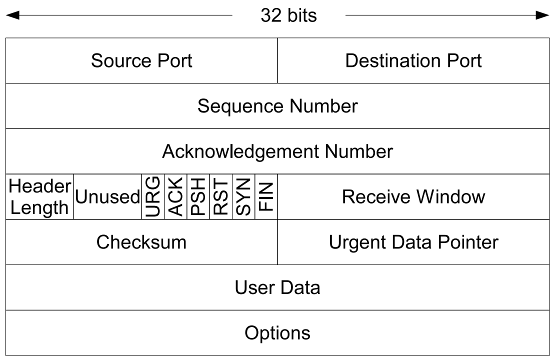

Fig. 2. TCP's segment format.

(UDP) and its well-known unreliable service. PR-SCTP is used to support real-time applications by relaxing retransmission guarantees [3]. Unlike TCP and UDP, however, SCTP provides support for multihomed end-points, i.e., devices with more than one network interface.

A. SCTP Basics

The segment definitions of contrasting transport layer protocols, SCTP and TCP, are provided in Figs. 1 and 2, respectively. Although there are many subtle differences between protocols, the most glaring is SCTP's compound segment structure. A common header, providing only basic control information, allows SCTP to differentiate segments by function. Compared to TCP's user data field, each SCTP common header is concatenated with one or more user data fields (known as chunks). Each chunk can carry either control or data information to an associated application. This architecture reduces overhead and increases efficiency by bundling smaller messages together. Each chunk, moreover, classifies a segment in terms of function. Out of the possible 255 different chunk types, RFC 4960 defines 14; while another 4 are reserved by the IETF for extensions, leaving 237 available for customization.

SCTP's message oriented approach also increases utilization from the use of selective acknowledgements (SACKs). SACKs clock out reordered segments and use the available congestion window more efficiently. TCP, on the other hand, tracks segments according to a sequential byte stream, therefore available network resources remain unused until cumulative acknowledgements (CUMACKs) are received.

Another difference is SCTP's two-tiered reliability system. At one level, transmission sequence numbers (TSNs) are used to maintain reliability and manage congestion control; while at another, stream sequence numbers (SSNs) serve as a mechanism for preserving stream independence. A stream may be allocated by an application as a means of dividing data into separate groups. For example, a web server may want to deliver multimedia data using one stream, while text uses another. If only one stream is employed, as is the case with TCP, head of line (HoL) blocking can inhibit one type of data (e.g., text) while another (e.g., video) is downloading. Traditionally, the application layer has had to maintain many TCP connections, simultaneously, to avoid HoL blocking.

Turning to segment dynamics, we see that SCTP establishes associations using a four-way handshake, as opposed to TCP's three-way approach. The four-way handshake attempts to mitigate "SYN attacks", common to TCP, by replying to a client's INIT message with a cookie composed of security information only the server can verify. The client echoes the cookie and upon receipt of the final echo, a connection is established. The threat is thwarted by allocating association resources (i.e., memory) only after the initiator confirms the connection request.

During data transfer, each chunk is assigned a TSN and may be bundled with other chunks into a single segment. Messages larger than the maximum transmission unit (MTU) are fragmented into multiple data chunks and delivered separately. At the receiver, data chunks are ordered and reassembled using SSNs/TSNs and begin/end flags, respectively. In contrast to TCP's cumulative acknowledgements, SCTP sends back selective information (i.e., gap acknowledgement blocks) so congestion control mechanisms can respond to TSNs that have been lost or received out-of-order. SACKs offer more information, albeit more bits per segment.

The dynamics of an SCTP association are displayed in Fig. 3. This shows the establishment, data transfer, and shutdown of a normal client-server SCTP *association* (analogous to a TCP connection). Additional overviews of SCTP can be found in [4]–[6].

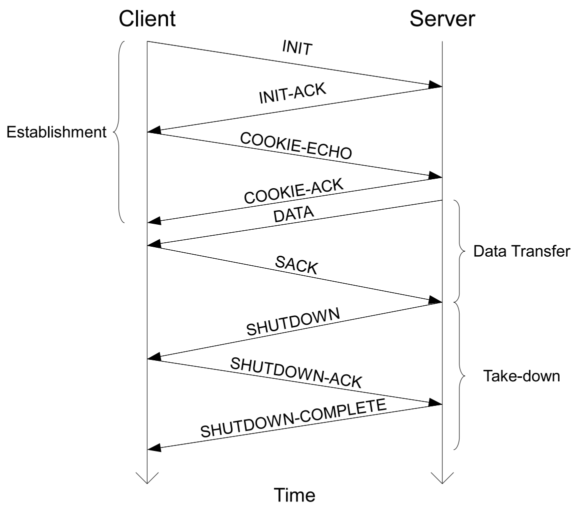

Fig. 3. SCTP's association dynamics.

B. SCTP Multihoming

Multihoming enables two hosts to establish a logical connection over a set of multiple network interfaces uniquely identified by separate IP addresses [4]. Typically, this type of support is applied as network-level fault tolerance or as middleware for seamless vertical handovers, i.e., switching data streams from one network technology to another without interrupting the application layer.

Currently, SCTP uses multihoming only as a backup service, that is, when an IP address becomes unreachable, SCTP will attempt to recover by sending new data to a secondary IP address. SCTP maintains an active list of alternate destination addresses by either sending periodic *heartbeat* probes or receiving acknowledgements from retransmitted data. Note that SCTP can employ a retransmission policy [7], e.g., one that retransmits lost segments to an alternate destination.

As previously stated, SCTP provides mechanisms for both flow and congestion control. Yet flow control is on a per association basis, and congestion control is managed per IP address. This is necessary to support multihomed interfaces with varying performance capabilities. Fig. 4 illustrates a typical sender/receiver pair in a multihomed environment using two different network interface technologies (e.g., Wi-Fi and Ethernet). Although the image depicts the sender with only one interface, this is not always the case. For example, it is possible for two multihomed clients to use SCTP for voice communication; while in transit, each client may switch access technologies any number of times.

C. SCTP Mobility

For a long time the Internet was uniform and static in nature, so legacy transport layer protocols, like TCP and UDP, never really had mobility in mind. Today, however, networks thrive on diversity and have evolved with an ability to alter their topological representation on-demand. Unfortunately, even with the development of SCTP, mobility was not a functional requirement. Not only must the transport layer negotiate available network resources for the application layer, it also needs

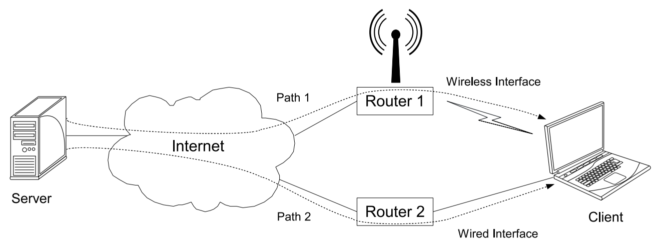

Fig. 4. A multihoming scenario.

to support network layer dynamic reconfiguration capabilities. Network reconfiguration occurs often when wireless devices move from one base station to another, or sometimes even when a link fails (e.g., cut, unplugged, overly congested). Although much of these adjustments may remain hidden from the transport layer, if the IP address changes, the application will lose its current session.

The popular choice for network mobility has been the Mobile IP (MIP) standard [8]. Unfortunately, a major drawback of MIP is that it does not support connection migration at the transport layer, even though many situations find this limitation undesirable: e.g., the Internet service provider updates a host's IP address; a new plug and play (PnP) network card is added to a multihomed server; or more commonly a mobile user moves from one subnet to another. In each of these scenarios, a service disruption would pause communication while the current association was torn down and reconfigured.

Although various research efforts have explored the area of transport layer mobility [9], dynamic address reconfiguration (DAR) [10] is a simple yet practical solution to SCTP mobility. DAR provides an SCTP association with support for three new features: (1) dynamic addition of IP addresses, (2) dynamic deletion of IP addresses, and (3) primary address update requests. The extension also defines two new chunk types, address configuration change chunk (ASCONF) and address configuration acknowledgement chunk (ASCONF-ACK). While the ASCONF chunks are used to either add/delete IP addresses or change the primary path, ASCONF-ACK chunks simply respond with success or failure messages to ASCONF requests.

A simple example of a mobile client associated with a static server across the Internet is shown in Fig. 5. The figure illustrates a mobile client moving from one subnet to another and the necessary DAR messages exchanged between both endpoints to keep the connection alive. At time (1), the mobile establishes a network presence in subnet 2, then informs the server of its additional IP address at time (2). Next, the primary path is updated to the new IP address, while the older one is removed from the association, (3)-(4). Although this is an intuitive solution for the transport layer, the lower layers (e.g., network, link, physical) may play a more complex negotiation game before any ASCONF chunks are transmitted.

DAR together with SCTP, is commonly referred to as mSCTP (or mobile SCTP). Several other variations of mSCTP include: cSCTP, SIGMA, and mSCTP+ [11]–[13]. It should be noted, though, that no version of mSCTP offers a completely decentralized system. For example, to initiate communication with a mobile that continually changes its address, an endpoint needs to know where to look first. The location manager,

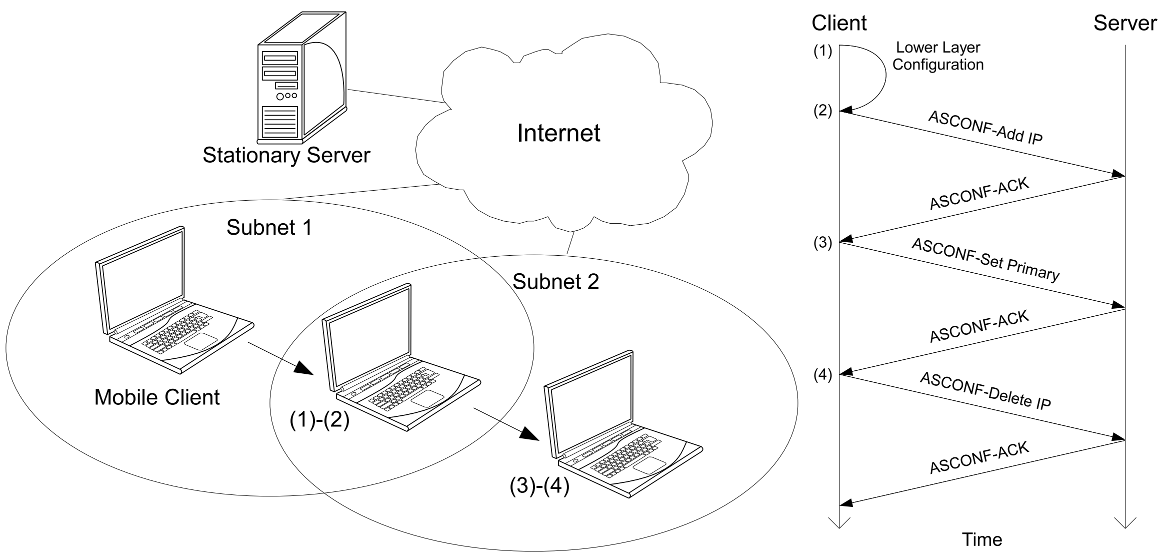

Fig. 5. Simple multihoming example using DAR.

like the name suggests, is a network entity that locates mobile clients. The location manager is statically stationed and keeps real-time address information on behalf of its mobile client, but is not necessarily bound to any particular network its corresponding client may be attached to.

III. Problems, Issues, and Challenges

The purpose of this section is to brief the reader on the most salient issues affecting SCTP multihoming, that is, handover management, concurrent multipath transfer, and cross-layer activities. In addition sub-problems related to these issues are introduced and their importance explained. Later in Section IV, a review of solutions to these sub-problems (proposed in the literature) is presented.

A. Handover Management

Handover offers the wireless client mobility by keeping connections alive while traversing access points. Multihoming expands that mobility by adding a wider range of access networks. To understand the role of handover in transport layer multihoming, we have created three new subcategories: (1) preemptive path selection, (2) fault tolerant path selection, and (3) post handover synchronization. In each subcategory, we explore an integral problem inherent to handovers at the transport layer.

1) Preemptive Path Selection: A typical multihomed environment will present the transport layer with a number of destination paths with disparate performance capabilities. Among these paths, SCTP must choose one for communication. Often, an end-point will want to choose the "best" path; providing either best-effort or quality of service (QoS) guarantees. Usually metrics like bandwidth, delay, and jitter are adequate, but if the multihomed environment consists of wireless links, signal strength is also critical. Given the range of networking applications, e.g., instant messages and video chat to name a few, and including other constraints, such as usage fees and power consumption, the "best" will not always be a *one size fits all*. Clearly then, multihoming increases handover complexity, by introducing a range of decision problems with a multitude of variables to explore.

2) Fault Tolerant Path Selection: SCTP avoids the path selection problem by letting a destination become unreachable (i.e., fail) before choosing an alternate. Without direct access to the link layer, however, SCTP has no way of knowing whether a fault has occurred. For example, consider a client end-point downloading a file using one of two interfaces. If the client's active interface loses its gateway connectivity (e.g., moves out of range of a base station), the server end-point will continue sending data—unknowingly—to a broken link. Notifying the server of connection loss (through the secondary interface) is trivial, but what happens when the mobile is slowly losing connectivity? For example, the wireless link is active but segment losses are increasing? Furthermore, what if the active interface already offers the highest data rate? Should the transport layer switch to a slower rate for better stability, or stay with its current incumbent for higher throughput potential? It is complicated questions like these that are the main reason why SCTP relies solely on fault tolerance for handover support. Nevertheless, multihoming should make losses more avoidable and interruption delays nonexistent.

3) Post Handover Synchronization: Whether handover is invoked from proactive (preemptive) or reactive (fault tolerant) path selection, following transition the transport layer is still susceptible to inefficiencies. *Spurious retransmissions*, brought on by segment reordering, causes the transmission rate to drop significantly. Reordering typically occurs from segments being delayed along a slow path while others are ushered ahead over a faster one. An undesired consequence of reordering is illegitimate loss indications prompting congestion window cutbacks. Without direct access to the link layer, the transport layer blindly infers segment loss by evaluating the order of arrivals. If too many segments are received out-of-order, the transport layer assumes loss and makes concessions to expedite recovery.

B. Concurrent Multipath Transfer

As mobile users manoeuvre through heterogeneous access networks, at times they may be presented with more than one access point with disparate performance capabilities and economic costs. For example, while one network may provide a large coverage area over one that is smaller, it may also bare higher costs. Furthermore, power consumption may also reduce battery lifetime depending on base station proximity or media access control (MAC) specifications. But if these caveats are of no concern, playing some sort of complex network selection game will only add overhead. Interestingly enough, however, when a device is multihomed, it is theoretically possible to use multiple network technologies simultaneously to aggregate throughput potential. The research community currently refers to such simultaneous transmissions as concurrent multipath transfer (CMT), but has also used terms like bandwidth aggregation, resource pooling, inverse multiplexing, load sharing, and even striping in similar contexts. Generally speaking, if more network resources (i.e., communication channels) were present, one would naturally presume an increase in upload speeds. Unfortunately, this has not been the case with SCTP for a number reasons: (1) unnecessary fast retransmissions; (2) crippled congestion window growth; (3) superfluous network traffic; (4) receive buffer blocking; and (5) naive scheduling.

1) Unnecessary Fast Retransmissions: Fast retransmission causes the sender to interpret network congestion, thereby reducing its congestion window. Moreover, a SACK with a gap report does not necessarily imply segment loss—that is to say, missing segments could be delayed and thus reordered. Still, SCTP infers a segment to be lost from additional SACKs because the probability of loss increases with every gap report received. Even in a lossless system, however, CMT reorders segments on a constant basis—diminishing connection quality in the absence of loss.

2) Crippled Congestion Window Growth: Another side-effect of CMT is overly conservative congestion window growth. When SACKs are received by the sender, they contain only new gap reports but no CUMACKs. Since it is the CUMACKs which grow the congestion window, previous gap reports that have already acknowledged segments will have no impact on how the congestion window grows. Even though segments may have arrived in-order, this behaviour will limit congestion window growth and prevent bursts of new data.

3) Superfluous Network Traffic: The next inefficiency caused by CMT is in regard to additional network traffic. Both SCTP and TCP reduce acknowledgement traffic by delaying acknowledgements until at least two can be sent together. Segments that are received out-of-order, rather, should immediately invoke reordered acknowledgement transmissions [1], [14]. Since a CMT receiver must frequently reorder segments, it does not delay acknowledgements, thus increasing acknowledgement traffic gratuitously.

4) Receive Buffer Blocking: Realistically we cannot assume a receiver will have infinite buffer space. In fact, mobile devices have very limited memory to begin with. Constrained buffers pose an even greater problem if different paths have disparate bandwidth characteristics. A sender is allowed to send only when the receive window, i.e., the amount of free buffer space, is greater than zero. Furthermore, the receive window increases only when segments are received in-order and passed to the application layer. CMT reorders segments, regularly, especially when paths are disparate in nature. The result is a temporary block on segment transmissions. When the receive window is full, blocking is devastating because it lowers network utilization and ultimately disables throughput.

5) Naive Scheduling: Simultaneous transmissions over paths with disparate bandwidth characteristics will not yield synchronous receptions. This environment will hinder the performance of a protocol offering ordered data delivery because higher sequence numbers will be received before lower ones. Furthermore, a simple round robin approach to scheduling segment transmissions over multiple paths will undoubtedly lower throughput because out-of-order segments will have to be buffered at the receiver's application queue. Even in the absence of loss and constrained buffer size, CMT needs intelligent scheduling to increase throughput and lower receiver side queueing.

C. Cross-layer Activities

This last area relates the preceding subsections by looking more closely at some of the additional responsibilities multihoming has imposed on the transport layer.

1) Bandwidth Estimation: Choosing the "best" path usually requires measuring bandwidth, delay, or jitter to provide best-effort or even QoS guarantees; how to measure bandwidth, delay, or jitter is an entire problem unto its own. Like TCP, SCTP will find difficulty in gathering accurate path measurements since it lies, detached from intermediary physical links, at the end-points of a connection. Moreover, if a metric can be gauged there is no guarantee it will remain accurate for any length of time since a network path is shared among many different sources with unknown traffic patterns. A conflict of interest can also arise if the transport layer has to send too much traffic into the network just to determine capacity. Each SCTP source needs to manage its own traffic efficiently so as not to overwhelm the entire network. While it may use the continuous flow of data and acknowledgements to infer bandwidth over the primary path, currently SCTP can only measure the capacity of its secondary paths from periodic probing.

2) Wireless Error Notification: When the network path is composed of wireless and wired links, the cause of each loss is more difficult to discern. It is common practise to assume wired links will not experience transmission errors, but may lose segments due to buffer overflows. On the other hand, while buffering is still a problem for wireless domains, the effects of signal fading, interference, noise, or Doppler shifts can corrupt segment transmissions. Since transport layer protocols are designed only to handle network congestion, losses over wireless channels are ambiguous. Due to this ambiguity, an end-point may not be able to discern congestion from wireless losses, leading to mismanaged resources and poor performance.

3) Network Intelligence: Routing is usually reserved for network layer protocols like Intermediate System To Intermediate System (IS-IS) and Open Shortest Path First (OSPF).

Assuming the system's terminating point is its IP address, protocols like IS-IS and OSPF work fine; but if the actual terminating point lies beyond the IP address, like in a multihomed system, some inefficiencies may go undetected. While the network layer sees a system of independent links between IP addresses, the multihomed transport layer sees only IP systems (i.e., pairs of IP addresses), where each IP system may or may not share common links. This raises issues of ambiguity, and challenges the notion that each IP system is truly independent. Similar to path selection, found in the handover management problem, the multihomed transport layer will have additional routing responsibilities, regardless of mobility.

IV. Solutions, Strategies, and Techniques

This section highlights the research efforts used to address the main issues surrounding SCTP multihoming. Note that each solution approach corresponds to a sub-problem with the same heading introduced in Section III.

A. Handover Management

1) Preemptive Path Selection: Making handover decisions or selecting an access network is a complicated problem. Many sophisticated strategies, such as economic cost functions or even machine learning techniques like fuzzy logic and neural networks have been used extensively to make optimized handover decisions. In fact, much of the work in this area has already been reported. For a comprehensive survey of vertical handover decision algorithms, we refer the interested reader to [15]. Vertical handovers involve reconfiguring an end-point's ingress access from one network technology to another. In [15], the handover problem primarily focuses on choosing the "best" access network given a set of user constraints. When comparing algorithms, handover performance can be evaluated by delay, number of handovers, handover failure probability, and throughput. We also direct the reader to [16], another survey that offers supplemental material on handover classification and mechanics.

Path selection strategies not mentioned in previous surveys include: SIGMA and ECHO. SIGMA, or Seamless IP diveristy based Generalized Mobility Architecture [12], is actually a mSCTP framework. SIGMA offers mobility to multihomed wireless devices through transport layer handover, location management, and simple path optimization. We say simple because the path selection tool compares only one variable, that is, received signal strength (RSS). SIGMA is, however, proactive in terms of path selection, because it sets no threshold, but rather initiates handover when an alternate interface has better signal strength.

Alternatively, ECHO, or endpoint centric handover [17], uses a more sophisticated path selection strategy. ECHO's primary objective, is to offer QoS guarantees for voice over IP (VoIP) calls using PR-SCTP. ECHO employs the E-model [18], an International Telecommunication Union (ITU) planning tool, to estimate the quality of end-to-end connections. Based on subjective testing, the E-Model rates the quality of a connection on a scale from 0 to 100. The technique transforms individual transmission parameters (e.g., signal strength, delay, jitter) into different *impairment factors* and adds them together. The output of the E-Model, R, is calculated by

$$R = Ro - Is - Id - Ie + A, \tag{1}$$

where Ro is the basic signal-to-noise ratio, Is represents impairments that occur simultaneously with voice encoding, Id sums all impairments due to delay, Ie gives the effects of equipment, and A is an advantage factor. The last variable, A, compensates for certain impairment factors by attempting to *level the playing field* (e.g., a willingness to accept lower quality connections under wireless conditions). Through a series of assumptions and simplifications, ECHO reduces the E-Model to

$$R = 93.34 - Id - Ie, \tag{2}$$

where the terms Id and Ie then correspond, more specifically, to delay/jitter and loss, respectively. ECHO regularly measures Id and Ie by transmitting the same data over each available path; then chooses the path with the best R score.

2) Fault Tolerant Path Selection: By foregoing the complicated phase of preemptive path selection, an easier way to make handover decisions is to simply wait until it is absolutely necessary, particularly, when the network tells you to. There are two major problems with this logic: (1) communication interruption, and (2) avoidable losses. The first problem occurs from simply waiting until it is absolutely necessary to update the transmission path. If we have to react to a problem, we will undoubtedly incur some kind of delay. But we can exacerbate that delay by choosing the wrong end-point to manage faults. For instance, the sender uses a parameter called Path.Max.Retrans (PMR) to determine a destination's reachability. Although failover (i.e., handover due to failure) is considered an implementation specific issue [1], the current recommendation is to compare the number of consecutive failed retransmission attempts to PMR. Then if this number exceeds PMR, the primary path should be considered inactive and an alternative will be selected. Since the retransmission timeout (RTO) increases exponentially following each consecutive loss, the total time necessary to detect a path failure can be expressed as $\sum_{i=0}^{\mathrm{PMR}} 2^i \mathrm{RTO}$. RFC 4960 suggests the following default settings: $\mathrm{RTO_{min}} = 1$ s, $\mathrm{RTO_{max}} = 60$ s, and PMR = 5. An SCTP association can then expect to experience a failover delay (i.e. the time from a link failure until a primary path update) to be anywhere between 63 and 360 seconds.

Staying with sender-side failure detection, some studies have tried to mitigate failover delay by monitoring the nature of loss events. The scheme proposed in [19], Sending-buffer multicast-aided retransmission with fast retransmission (SMART-FRX), works in three ways. The first is a minor change to the retransmission policy; lost segments notified by missing reports are always retransmitted to the same address, and those that have timed out are sent to an alternate destination. This is attributed to the assumption that losses from missing reports are due to network errors rather than poor conditions; if an alternate path has lower bandwidth over the primary path, a delay due to retransmission may inadvertently reduce throughput even further. The second method avoids handover loss by simultaneously transmitting the same data over multiple paths during periods of instability. Finally, the third approach mitigates failover delay by monitoring loss events. If at some point the number of consecutive retransmissions on the primary path exceeds PMR, the primary path becomes inactive and the alternate takes over. If, however, a retransmission over the primary path is successful before exceeding PMR, no change will take place.

The multipath transmission algorithm (MTA) is a method aimed at reducing losses while the behaviour of the primary path begins to experience poorer conditions [20]. Avoiding any congestion window and reordering problems, the authors direct their work toward real-time applications and simply disable the congestion control and fast retransmission mechanisms. Similar to [19], MTA also transmits the same data over multiple paths during periods of instability. The algorithm implements two transmission modes: *singlepath* and *multipath*. During singlepath, the source sends information only along the primary path, while multiple copies of the same information are sent along an alternate path during multipath mode. Multipath mode starts when the number of consecutive retransmissions to the primary path reaches Multipath.Threshold (MT), such that MT $<$ PMR. In multipath mode, the sender monitors the stability of the new path by sending heartbeat segments. When the sender receives two consecutive acknowledgements for heartbeat segments, a new counter is increased by one and compared to a parameter called Stability.Threshold (ST). If this new stability counter exceeds ST before the number of consecutive retransmissions overtakes PMR, the alternate path will become the new primary path.

The two most basic operations required for transport layer handover are: (1) adding a new IP address, and (2) changing destination paths. When to exploit these operations now becomes the quintessential question. An experiment from [21] describes a dualhomed SCTP receiver crossing randomly between two access points. At the same time, a file is being downloaded from a stationary server across the Internet. Transport layer *Add-IP* and *Path-Change* functions are initiated when preset RSS thresholds are broken. Analysis of this study focused on the effect of signal strength for each threshold. For example, both aggressive and conservative thresholds were chosen for either Add-IP or Path-Change operations, where the terms aggressive and conservative refer to lower and higher threshold levels, respectively. In terms of Add-IP, the aggressive rule adds a new IP address when the signal strength reaches some minimum threshold. On the other hand, the conservative rule only adds a new IP when the strength is greater than that of the current IP address. The results of the experiment showed using an aggressive Add-IP with a conservative Path-Change will yield the fastest download times. Clearly, download times will be faster under the aggressive Add-IP rule since the alternate path is available sooner. But higher throughput from the conservative Path-Change rule may not be as apparent. It was said that by making frequent path changes, as with the aggressive rule, the association becomes more unstable, hence lower performance. With respect to this type of scenario, a hysteresis threshold (i.e., introduced state-change memory) may be able to mitigate undesirable connection oscillation between access networks [22], [23].

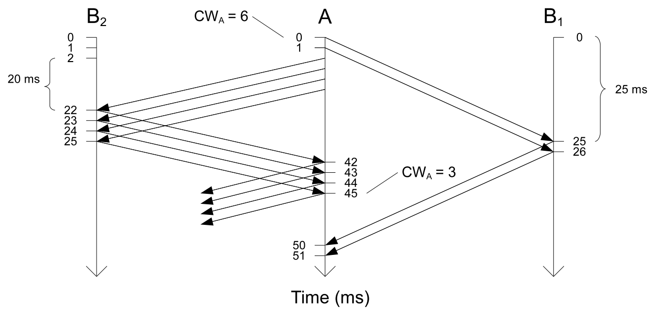

Fig. 6. Congestion window cutbacks and spurious retransmissions during handover.

Before moving on, we would like to point out an intrinsic vertical handover problem. Again, assume two end-points share a SCTP association across the Internet. Currently, the wireless multihomed receiving end-point finds itself moving from one base station to another and between subnets. Although the primary path may use either of the two available destination IP addresses while the receiving end-point is in range of both base stations, when the signal becomes so low that a connection cannot be maintained, the primary path of the SCTP association will have to be updated to acknowledge the receiving end-point's only location of network access. The question then is, "At what point should the primary path be updated?" While analyzing this problem, we may first want to define what our main goal is, e.g., is it more important to maximize throughput to transfer data as fast as possible or perhaps minimize loss so that a stable connection is retained during handover? We then might be interested in such things as bandwidth (i.e., data rate) and whether the receiver is entering a subnet where the bandwidth will be higher or lower compared to the old subnet. In either event, the state variables controlling SCTP will need to be monitored carefully.

3) Post Handover Synchronization: Network capabilities will often change after handover; sometimes for the better and other times for the worse. Regardless of the outcome, synchronization issues at the transport layer can lead to undesirable results. A handover example describing the effect of reordered segments is shown in Fig. 6. In this example end-point A is sending to multihomed end-point B using address 1 (i.e., B_1). The propagation delay from A to B_1 is 25 ms, where all other delays (i.e., transmission, queueing, and processing) are considered negligible. Furthermore, the one way delay from A to address 2 of B (i.e., B_2) is 20 ms. At time 0 the congestion window is 6 segments. After sending two segments, and at time 2, the primary path is updated and the remaining four segments are sent to B_2. Since it takes longer for the first two segments to reach B over B_1 than it does the remaining four over B_2, when the acknowledgements for the last four reach A, the fourth missing report for the first two segments will trigger a fast recovery due to the higher than expected sequence numbers. This causes the congestion window to be halved and retransmissions made at time 45, albeit unnecessary, since the first two segments will be acknowledged at times 50 and 51, respectively. Following the loss indication, the spurious retransmissions made by the sender only add to the problem by further congesting the network with redundant data.

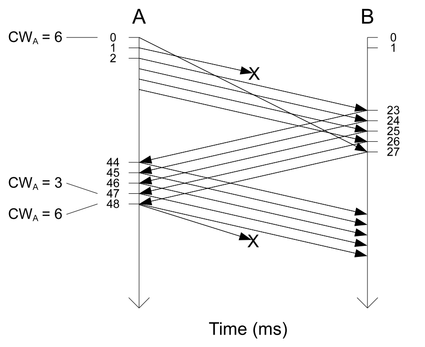

Fig. 7. The Eifel problem.

Ideally, we would like to avoid illegitimate loss indications and do away with spurious retransmissions altogether, but finding a solution is no easy task. Currently, techniques work only on the principle of spurious retransmission detection. For example, the Eifel algorithm [24] informs a sender of premature retransmissions after receiving acknowledgements for segments sent before retransmissions. Following a loss indication and retransmission of any missing segments, the Eifel algorithm simply compares the transmission time of all acknowledged segments received before the retransmitted ones. If an acknowledgement is received for a segment with a transmission time before the loss indication, a spurious retransmission is detected and congestion control is reverted back to its previous state.

Although the simplicity of the Eifel algorithm can mitigate some performance degradation due to spurious retransmissions with prompt detection, it suffers from an inability to differentiate between reordering and actual congestion loss. An example of such is shown in Fig. 7. The example shows a delayed segment transmitted at time 0 followed by a lost segment at time 1. When the fourth missing report arrives at time 47 a retransmission is made on behalf of the first segment, albeit at time 48 an acknowledgement for the first segment arrives. Depending on the retroactive policy for spurious detection, using the Eifel algorithm could return SCTP to its original congestion control state, which may result in another loss due to congestion, all the while disregarding the fact that the second segment was actually lost.

Another strategy, known as duplicate SACKs (DSACKs) [25], will wait for all retransmissions to be acknowledged before concluding whether any or all were indeed spurious. But again the system must make spurious retransmissions before determining if they were warranted or not. This can be likened to the idea of *throwing an alleged witch into a lake only to see if she floats*. Ladha et al. propose the combination of Eifel and DSACK to combat spurious retransmissions due to reordering in [26]. Fig. 8 provides a modified flowchart of the combined process highlighting the necessary steps to determine spurious retransmissions with respect to the functional requirements proposed in [26]. Note that by testing whether a CUMACK is greater or equal to the highest retransmitted segment implicitly concludes that the original transmission time is also less than the send time of the first retransmission.

Presented in [27], SCTP handover optimization (SHOP) is a method for reducing segment reordering after handover. SHOP

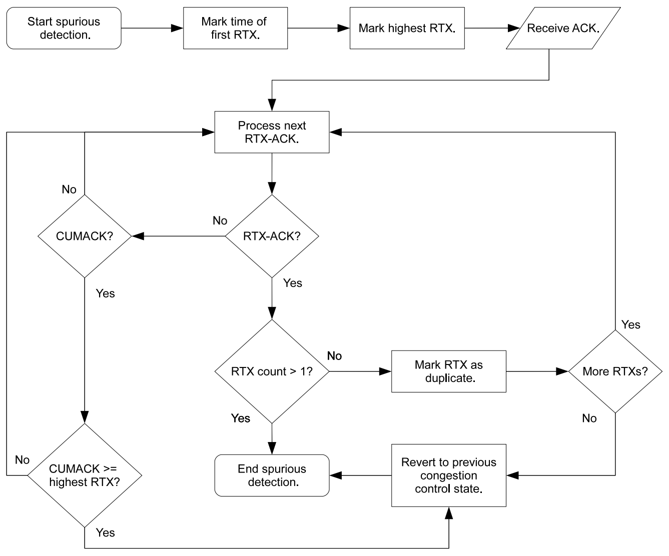

Fig. 8. Eifel/DSACK algorithm.

is a proactive scheme that monitors the average RTT of any impending handover path. If the new path has a lower latency over the old one (i.e., $\Delta\text{RTT} > 0$), then following handover all queued transmissions are immediately postponed. SHOP simply backs off for a period of $\delta \cdot \Delta\text{RTT}$ until ordered data arrivals can be better approximated. The coefficient δ is a connection parameter applied to avoid inaccurate estimations due to receiver side delayed acknowledgements.

A final proposal called handover retransmission for mSCTP (HR-mSCTP) is presented in [28]. The proposed work also sends redundant segments to prevent congestion window cuts due to illegitimate loss indications, or timeouts following handover. By preemptively retransmitting all unacknowledged segments before sending new ones along the alternate path, the authors guarantee the congestion window will not be affected by spurious retransmissions.

B. Concurrent Multipath Transfer

1) Unnecessary Fast Retransmissions: Iyengar et al. solve the spurious retransmission problem for CMT with their Split Fast Retransmit (SFR) algorithm [29]. SFR validates the legitimacy of missing reports by analyzing the destination address of reordered segments; specifically, SFR tracks the highest acknowledged TSN for each destination address, indicated by the variable *highest*. As an example, if TSNs 1 through 5 were sent to destination 1 and TSNs 6 through 10 were sent to destination 2 and the receiving SACK carried TSN acknowledgements 4 and 6; then *highest* for destination 1 would be 4, while *highest* for destination 2 would be 6. We have provided a simplified version of the SFR algorithm in Fig. 9.

2) Crippled Congestion Window Growth: Another algorithm in [29], Cwnd Update for CMT (CUC), allows the congestion window of each destination to grow independently. Instead of using a single CUMACK variable, CUC employs individual CUMACKs for each destination. After each transmission, CUC implicitly notes the expected order of TSNs for each destination address. Again, if TSNs 1 through 5

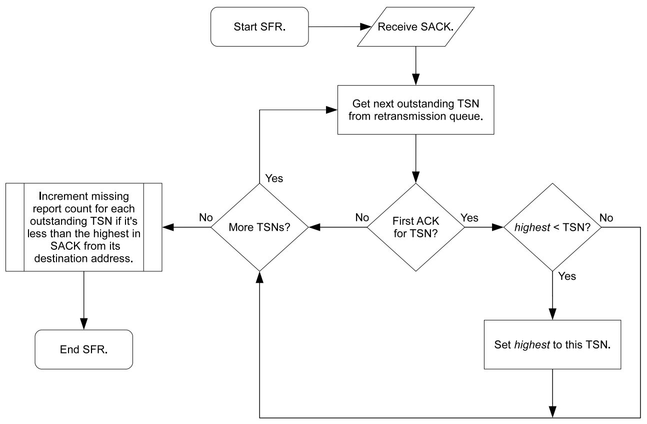
Start SFR.
Receive SACK.
Get next outstanding TSN from retransmission queue.
Yes
Increment missing report count for each outstanding TSN if it's less than the highest in SACK from its destination address.
No
More TSNs?
No
First ACK for TSN?
Yes
highest < TSN?
No
Yes
End SFR.
Set highest to this TSN.

Fig. 9. The Split Fast Retransmit Algorithm.

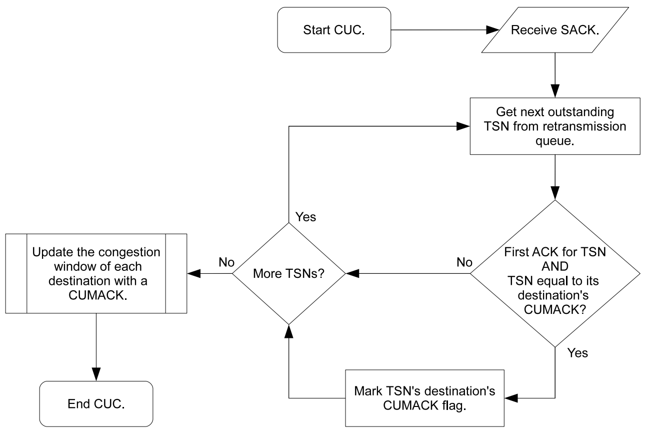
Start CUC.
Receive SACK.
Get next outstanding TSN from retransmission queue.
Yes
Update the congestion window of each destination with a CUMACK.
No
More TSNs?
No
First ACK for TSN AND TSN equal to its destination's CUMACK?
Yes
End CUC.
Mark TSN's destination's CUMACK flag.

Fig. 10. Cwnd Update for CMT.

were sent to destination 1 and TSNs 6 through 10 were sent to destination 2 and the receiving SACK carried TSN acknowledgements 4 and 6; then, a CUMACK would update the congestion window on destination 2 but not on 1. An interpretation of CUC is provided in Fig. 10.

To be complete, we should also mention the independent per path congestion control for SCTP (IPCC-SCTP) algorithm [30] as well as Load-Sharing-SCTP (LS-SCTP) [31]. IPCC-SCTP and LS-SCTP will also prevent spurious retransmissions and poor window growth by assigning each TSN a unique path sequence number (PSN). Upon reception of a SACK, they will perform similar operations as SFR and DAC by marking only those TSNs as received or missing if their corresponding PSNs are either in or out-of-order, respectively. Still, IPCC-SCTP incorporates an implicit extension to the sender while adhering to the SCTP standard; LS-SCTP requires changes to both the sender and receiver as well as reformation of the SCTP segment.

3) Superfluous Network Traffic: To curb the barrage of redundant acknowledgements from reordered segments, a routine called Delayed Ack for CMT (DAC) was incorporated into SFR [29]. DAC first mandates a delayed acknowledgement rule regardless of gaps. Then if a SACK acknowledges TSNs sent to the same destination as one from the retransmission queue, it should increment its missing report count by the number of segments received prior to the previous SACK, i.e., 1 or 2 segments.

4) Receive Buffer Blocking: A study revealing the effects of constrained buffers on CMT is showcased in [32]. The authors demonstrate poor transfer times when buffer space is shared between high and low bandwidth paths. In fact, in some circumstances using only the better of two paths actually achieves higher throughput. They hypothesize that repeated retransmission timeouts along the lower quality path degrades throughput. Moreover, it is their viewpoint that smart retransmission policies should mitigate receive buffer blocking. Various retransmission policies for CMT have been studied in both [32] and [33], respectively. Conclusions were vague, but the authors feel a retransmission policy that does not take the loss rate of each path into account will be unable to aid the receive buffer blocking problem.

One approach to receive buffer blocking is to schedule segment transmissions more intelligently—reducing the effect of reordering. The simplest CMT scheduling technique queues new segments in a round robin fashion as congestion windows become available [29], [32]. Unfortunately, since this method does not account for bandwidth nor delay across each path, it cannot be expected to minimize receive buffer blocking. Assuming the lowest transmission rate and propagation delay is known for each path, a more intelligent scheme could predict which path can deliver a segment earlier. The bandwidth aware scheduler proposed in [34], calculates the earliest delivery time of a segment by

$$C_i = C_i + \frac{D}{B_i} + P_i, \tag{3}$$

where C_i is the current delivery time of segments on path i, D is the segment size, B_i is the bandwidth (i.e., data rate) on path i, and P_i is the propagation delay. Although the receive buffer blocking problem is not studied explicitly, simulated file transfer times are shown to improve when using this method.

The receive buffer blocking problem can also be extended to path failures. Assuming the presence of more than one destination and following a timeout, the sender can avoid a second consecutive loss by retransmitting to an alternate destination. Even though this may mitigate the initial damage with prompt recovery, if a reactive failover scheme is employed, the sender will continue transmitting new data to a broken link until PMR reaches its threshold. During each failed attempt, receive buffer blocking will degrade connection quality. Moreover, if the RTO is exponentially increased after each consecutive loss, the blocking problem will suffer a similar delay. Noted in [35], this problem is tackled by introducing a new state called *potentially-failed* (PF). Following a timeout, the corresponding destination will be flagged PF. All new data is then transmitted over an alternate path and a heartbeat probe sent to the PF destination. If and only if a heartbeat is acknowledged will the PF destination be returned to a state of congestion avoidance.

5) Naive Scheduling: Optimal scheduling is achieved in [36]. The authors first model the problem of distributing data segments over multiple paths as a Markov chain. Then a Markov decision process (MDP) is formulated to specify a scheduling policy, specifically, which actions are taken given system state and time step. Following this, the On-line Policy Iteration (OPI) algorithm was proposed to approximate optimality. Although this work has substantial merit, it forgets two major constraints of the transport layer, that is, limited receive buffer and ordered data delivery. Without mention of these constraints, optimal throughput could be achieved, but unrealistically.

Similar to the receive buffer blocking problem, CMT can stall communication if a sender's transmission queue is constrained. Due to the possibility of reneging, strictly speaking, a receiver's ability to give an acknowledgement only later to disregard it, SCTP mandates a sender to maintain copies of out-of-order segments until a CUMACK asserts their reception. If one path is considerably faster than another and the transmission queue is limited, the sender may have to stop transmitting over the faster path while it waits for CUMACKs from the slower path. Yilmaz et al. [37] solve this problem with the creation of non-renegable selective acknowledgements (NR-SACKs). The new acknowledgement type simply tells the sender to remove acknowledged segments from its transmission queue, regardless of reordering.

Lastly, modifications to SCTP enabling CMT of multimedia traffic are explored in [3]. Employing a new adaptive scheduling algorithm, the authors propose a transport protocol called Westwood SCTP with Partial Reliability (W-PR-SCTP). Partial reliability is achieved by putting an upper bound on the number or retransmissions without acknowledgement. W-PR-SCTP is based on estimating the available bandwidth on each path and intelligently scheduling data chunks across paths. Using a similar method to TCP Westwood+ [38], W-PR-SCTP bases its path estimates on contiguous, non-overlapping time windows, taking the maximum of either one RTT, or 50 ms. Bandwidth is then measured using a smooth auto regressive moving average. Data chunks are scheduled and assigned to paths by computing a path's current *reception index*. The reception index of a path is calculated by dividing the accumulated size of the current chunk, any outstanding chunk(s), and any buffered chunk(s) by the estimated path bandwidth. Although W-PR-SCTP ignores the size of the receiver window by assuming an infinite buffer, simulation studies showed that W-PR-SCTP will out perform a naive or greedy scheduler (i.e., one that sends data over a path as soon as the congestion window is available) in terms of jitter and overall throughput.

C. Cross-layer Activities

1) Bandwidth Estimation: Due to similarities, the major techniques available to TCP for bandwidth estimation may also be used by SCTP, e.g., variable packet size (VPS) probing, packet pair/train dispersion (PPTD), self-loading periodic streams (SLoPS), and trains of packet pairs (TOPP) [39].

A combination of bandwidth estimates are used by Fracchia et al. to gauge the capacity of primary and secondary paths [40]. A simple TOPP approach is used on the secondary path by sending a train of six variable sized segments (two small, two large, and two small) instead of the regularly scheduled heartbeats. At the receiver, the dispersion times of the large sized segments are measured and returned to the sender via a heartbeat acknowledgement. The bandwidth of an alternate path can be calculated by simply dividing the size of either large segment by their corresponding separation time. Taking advantage of the continuous flow of data, bandwidth is estimated over the primary path from RTT and SACKs. A bandwidth sample is calculated from the k^{th} RTT by

$$B_k = \frac{D_k}{\Delta_k}, \tag{4}$$

where D_k is the reported number of bytes acknowledged by a SACK with a RTT of Δ_k. A low pass filter is then applied to average the samples so the mean bandwidth is

$$\hat{B}_k = \alpha \hat{B}_{k-1} + (1-\alpha) B_k, \tag{5}$$

where α is a constant, such that $\alpha \in (0, 1)$.

2) Wireless Error Notification: Although TCP and SCTP are experts at handling common congestion along wired links, they are unable to distinguish errors caused by fickle wireless channels. If these undesired conditions are only temporary, e.g., from a sudden change in phase, congestion control reactions will be unnecessary and overkill. Already, a number of surveys have reported various solutions for handling TCP over wireless links [41]–[43]. Again, due to the close relationship between the two, the research efforts for TCP may also serve as excellent resources for SCTP's plight.

As an example of recycled research, the sender-side solution, TCP Westwood+, is used by SCTP to discriminate between congestion losses and wireless errors in [40]. The process works by comparing the output rate, i.e., *cwnd*/RTT, with the most recent bandwidth estimate following a loss indication. If the output rate is larger than the bandwidth, congestion is inferred because all available bandwidth is being utilized. If this is not the case, however, a wireless loss is more likely, and instead of dropping the congestion window by half, it simply sets the slow start threshold to

$$ssthresh = \frac{B \cdot \text{RTT}_{\text{min}}}{D}, \tag{6}$$

where B is the bandwidth estimate, D is the maximum segment size, and RTT_{min} is the minimum observed round trip time. If four missing reports have triggered a loss indication, then the *cwnd* is set to *ssthresh*. We should point out that bandwidth probing, required by the process, can obscure capacity measurements—something not considered by the researchers in their study.

Another approach uses the link layer at a wireless interface to fragment a bundled SCTP segment into its individual chunks. The ideology is that smaller segment sizes should have lower probability of transmission corruption over larger ones. Presented in [44], the process works by disassembling and reassembling bundled SCTP segments between a wireless receiver and its access point. Assuming all disassembled chunks are transmitted in-order, when the last chunk is received a SACK will inform the sender if any were lost. Senders noting that a lost chunk had been fragmented, will infer wireless errors as opposed to congestion loss. The effectiveness of this process, however, lies in the assumption that SCTP segments have more than one chunk. If only one chunk were sent per segment, there could be no inference. Furthermore, unless the receiver knows when it should have received the last chunk of a disassembled segment, it will not know how long to wait before reporting the segment missing.

Finally, TCP's explicit error notification (ECN) [45] mechanism can also be applied to SCTP. ECN is implemented in two layers of the OSI model, i.e., the transport and network. At the network layer, routers determine congestion by comparing the current average queue length with some preconfigured threshold. When the router's queue length exceeds this threshold, it begins marking the ECN bit on all outgoing IP packet headers. Transport layer receivers employing ECN may then inform corresponding senders of congestion by setting similar flags in TCP acknowledgement segments. In turn, senders may then tune their output rates in a more informative manner. Intuitively, if loss indications are made without ECNs, then a wireless or noncongestive loss may be inferred. Since RFC 4960 has reserved two chunks for future ECN extensions, SCTP is poised to apply this well researched technique immediately. The work in [46] evaluates SCTP ECN and makes the following recommendations:

1) Do not use the *congestion window reduced* (CWR) chunk as it consumes more bandwidth than its TCP counterpart through additional overhead.
2) Only reduce send rate once per window of congestion losses (i.e., treat remaining losses as noncongestive and perform normal retransmission procedures).
3) To prevent underutilization, the minimum congestion window of an SCTP sender should be

$$cwnd_{\text{min}} = \frac{B \cdot \text{RTT}}{8} + \text{MTU}, \tag{7}$$

where B is the bandwidth of the path's bottleneck link and MTU is the maximum transmission unit.

3) Network Intelligence: Assuming bandwidth information is available and unchanging, SCTP multihoming may be able to gain a performance advantage using asymmetric path selection. An asymmetric path is one where the forward and reverse paths do not have equal one-way delays. In fact, Internet users often subscribe to asymmetric paths when purchasing data plans from Internet service providers (ISPs). Typically, a subscriber has higher download rates than they do for uploading. When an endpoint is multihomed, the captured one-way delays could result in a significant improvement if used efficiently. Take for example the illustration in Fig. 11. Here we have a multihomed client with two interfaces. While interface 1 has download and upload rates of 3 Mbps and 512 Kbps, interface 2 offers rates of 2 Mbps and 1 Mbps, respectively. Clearly, better performance will be achieved using only one interface to download and the other to upload, than using a single interface for both. Recommended in [47], each sender should maintain a matrix of delays in each direction to generate all RTT combinations and optimize path selection. Referring to our simple example and assuming a constant segment size, to achieve the best performance, the server should transmit segments to interface 1, and the client should send acknowledgements over interface 2.

Identifying shared bottlenecks is another strategy to make better use of system resources. Noted in [48], unique bandwidth estimates might be misleading if data streams share the same bottleneck link across separate network paths. Since the transport layer has a blind view of the network topology, the authors suggest comparing the autocorrelation in RTT delays of each path with the cross-correlation of combined paths. Then, two paths will share a common bottleneck if their cross-correlation is larger than the autocorrelation over one path. Upon this discovery, multiple transfers could be consolidated into one destination address—simplifying an SCTP association. Deallocating redundant interfaces, not only cures inefficiency, but also frees resources for alternative uses.

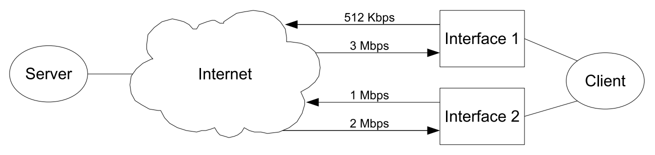

Fig. 11. Asymmetric path selection.

V. Discussion

We now offer our own analysis of SCTP multihoming; commenting on the surveyed techniques and making recommendations for future research.

A. Handover Management

Preemptive path selection matches multihomed devices with the "best" available access network; but at what cost? Mobile devices are notorious for poor processing power, little memory, and limited battery life. Furthermore, applications like VoIP require rapid decision making results, something optimization cannot always provide. With respect to these caveats, optimal path selection is probably a long way off, though, existing approximation techniques can achieve good results in a reasonable amount of time, specifically, meta-heuristics. We suggest the preemptive path selection problem make use of trusted meta-heuristics like simulated annealing, tabu search, or genetic algorithms. Meta-heuristics, moreover, often converge in a fraction of time their optimal counterparts might take. Nevertheless, each application will need to define an objective function for the transport layer to solve. ECHO is a good framework for preemptive path selection because it defines an objective, i.e., maximize R. ECHO also translates link layer metrics (e.g., delay, jitter, transmission rate) into problem specific parameters. Because many network applications can be grouped together (e.g., voice, video, data), a path selection sublayer (between the application and transport layers) may prove beneficial to the future demands of the multihomed communication system.

Fault tolerance reduces the path selection problem to a reactive service. This should be considered a valid approach because the transport layer is already complicated enough, and the limitations of mobile devices will find optimal path selection taxing. Due to slow failure detection and unreasonable communication delays, however, SCTP's failover guidelines are unacceptable to most applications. Furthermore, the current research strategies place too much emphasis on sender-side failure detection; especially when the channel information is at the receiver. Examining whether it is best to wait for the sender to initiate a primary path change after failover or have the receiver automatically reconfigure the connection based on its observed channel conditions, the data favours the latter. For instance, a number of articles have published results showing significant service disruption during SCTP based handovers [23], [49]–[51]. Using DAR, SCTP end-points can easily monitor their connection status and update the sender of path changes. Unfortunately, if the receiving end-point waits too long, communication will be disrupted and even segments may still be lost if the sender is uninformed at the exact moment of path failure.

The fault tolerant path selection problem becomes more complicated when mobile devices move away from high bandwidth access points. Even if the signal strength of a new base station is stronger than its predecessor, the mobile will likely want to remain connected to the old base station as long as possible. It is our recommendation that handover prediction play a part in solving this problem. In most cases, handover prediction is used by base stations to make advance resource reservations for calls with high handover probability (i.e., calls most likely to transition from one cell to another). To make valid predictions, the mobile is usually fitted with a transceiver which feeds coordinates to a satellite positioning system. These coordinates are then sent to base stations in close proximity to the mobile, where electronic road maps are employed to make best guess approximations on handover time and location. The wealth of research (e.g., [52]–[55]) in this area should pave the way for its assimilation with transport layer multihoming.

Moving the responsibility of path selection to the receiver should yield shorter interruption delays, but signalling will become more important if we want to maximize throughput. If a mobile finds itself crossing thresholds on a consistent basis, signalling and reconfiguration costs may only exacerbate the problem. A hysteresis can offer some form of stability, but should do so by adapting to environmental conditions. For example, in the same way RTO is tuned to the ever changing RTT, so should a hysteresis reflect the rate and trade-off of path switching.

Following handover, spurious retransmissions plague the entire communication system; while the network suffers from inefficient use, the end-point loses throughput momentum. So far, post handover synchronization is handled in one of two ways: (1) multipath transmissions (i.e., sending the same data over multiple paths), or (2) spurious retransmission detection. In either case, the system is treated through redundant measures. Similar to fault tolerant path selection, better prediction may also improve handover synchronization. By forecasting the change in connection quality (e.g., bandwidth, delay), SCTP should be more prepared and thus more capable of efficient handover management.

The techniques (from this paper) used for handover management are summarized in Table I. While many approaches share the same characteristics, they differ in their implementation benefits.

B. Concurrent Multipath Transfer

A number of solid contributions has elevated the operational efficacy of CMT. By curbing the effects of reordering and stunted congestion window growth, CMT can achieve aggregate performance gains. But substantial work is still necessary on the scheduling front. How to choose transmission paths to minimize receive buffer blocking is a challenging problem that needs to be addressed. The MDP approach can certainly aid heuristic design as long as transport layer constraints are kept in mind, but this may prove challenging.

Our own work focuses on the performance issues of CMT with bounded receive windows [56]. By intelligently scheduling the slower of two-path transmissions, we can maintain a continuous flow of new data without interruption; while at the same time approaching the theoretical limits of aggregated performance. Our scheduling algorithm works as follows: after a congestion window update, we estimate how many segments can be delivered over the faster path ahead of a single segment using the slower one; we then use this estimate to transmit the next segment (i.e., set of bytes) that will minimize buffering at the receiver. This is in contrast to the current scheduling policy, which simply sends the next sequential segment over each path regardless of performance disparity.

C. Cross-layer Activities

Without explicit feedback from Internet routers, SCTP end-points use RTT estimates to measure network performance capabilities. The various methodologies used by TCP can also be applied to SCTP. We should point out, however, the intrusive nature of measurement techniques; that is, segments always need to be transmitted before any valid results can be attained. This makes the network susceptible to gratuitous degradation and inefficiency. Fortunately, some measurement strategies are less intrusive than others—though they cannot always provide the right amount of information. Since most of the mulithoming techniques we have discussed assume the availability of bandwidth conditions, care must be taken each time the network is measured.

Losses occurring over wireless hops imply a very different result than those experienced over a wire. Wireless losses usually occur at random times and are often caused by unforeseeable events like interference or sudden movement. Of the three approaches applied to SCTP, two show significant merit, that is, TCP Westwood+ and ECN. Though concern for both exist; while the former suffers from intrusive bandwidth probing, the latter has scalability issues. For instance, all routers must employ ECN, and there may be 10s or even 100s of routers between an SCTP sender and receiver. Without a defining solution, differentiating between wired and wireless losses should be a primary concern for multihomed devices as we embark further into the age of wireless computing.

Identifying shared bottlenecks and routing traffic at ingress points are just some of the newer responsibilities assumed by the transport layer with the assimilation of multihoming. Needless to say, with more research in this area, SCTP will likely evolve to take on even more roles than its transport layer predecessors ever imagined.

VI. Conclusion and Final Remarks

In this article we have presented a comprehensive review of transport layer multihoming using the steam control transmission protocol (SCTP). Currently, the main areas of study include: handover management, concurrent multipath transfer,

TABLE I
SCTP HANDOVER TECHNIQUES

Algorithm Name	Problem Addressed	Solution Approach	Bandwidth Aware	Benefits	Drawbacks
SIGMA	path selection	maximize single variable function	no	application independent	best-effort
ECHO	path selection	maximize multi-variable function	yes	QoS	application specific
SMART-FRX	losses, failover delay	multipath transmissions, retransmission policy	no	loss differentiation	failover scheme, redundant segments
MTA	losses	multipath transmissions	no	simple	failover scheme, redundant segments
Eifel	spurious retransmissions	detection	no	fast	unreliable
DSACK	spurious retransmissions	detection	no	reliable	slow
SHOP	reordering	smart scheduling	yes	receiver initiated	monitoring overhead
HR-mSCTP	reordering, losses	multipath transmissions	no	receiver initiated	redundant segments

and cross-layer activities. A survey of strategies among research efforts has brought forth improved performance while in some cases even complete solutions.

Looking toward the future, we envision cross-layer activities to play a more substantial role in transport layer multihoming. Whether it's prediction techniques for handover management, or intelligent scheduling for CMT, unless careful consideration of many system variables (e.g., path capacity, estimated delivery times, receive buffer size, wireless channel conditions) are taken into account, inefficiencies will continue to plague the network. In any case, future applications will require even higher throughput than those of today, and multihoming using SCTP is one way to meet those demands.

REFERENCES

[1] R. Stewart, "RFC 4960: Stream control transmission protocol," *Request for Comments, IETF*, 2007.

[2] R. Stewart, M. Ramalho, Q. Xie, M. Tuexen, and P. Conrad, "RFC 3528: Stream control transmission protocol (SCTP) partial reliability extension," *Request for Comments, IETF*, 2004.

[3] M. Fiore, C. Casetti, and G. Galante, "Concurrent multipath communication for real-time traffic," *Comput. Commun.*, vol. 30, no. 17, pp. 3307–3320, 2007.

[4] R. Stewart and C. Metz, "SCTP: New transport protocol for TCP/IP," *IEEE Internet Comput.*, vol. 5, pp. 64–69, 2001.

[5] A. Caro Jr, J. Iyengar, P. Amer, S. Ladha, G. Heinz, and K. Shah, "SCTP: A proposed standard for robust Internet data transport," *Computer*, vol. 36, no. 11, pp. 56–63, 2003.

[6] S. Fu and M. Atiquzzaman, "SCTP: State of the art in research, products, and technical challenges," *IEEE Commun. Mag.*, vol. 42, no. 4, pp. 64–76, 2004.

[7] A. L. Caro, P. D. Amer, and R. R. Stewart, "Retransmission policies for multihomed transport protocols," *Comput. Commun.*, vol. 29, no. 10, pp. 1798–1810, 2006.

[8] D. Le, X. Fu, and D. Hogrefe, "A review of mobility support paradigms for the Internet," *IEEE Commun. Surv. Tutorials*, vol. 8, no. 1, pp. 38–51, 2006.

[9] M. Atiquzzaman and A. Reaz, "Survey and classification of transport layer mobility management schemes," in *Proc. IEEE Intl. Symposium on Personal, Indoor and Mobile Radio Communications*, vol. 4, 2005, pp. 2109–2115.

[10] R. Stewart, Q. Xie, M. Tuexen, S. Maruyama, and M. Kozuka, "RFC 5061: Stream control transmission protocol (SCTP) dynamic address reconfiguration," *Request for Comments, IETF*, 2007.

[11] I. Aydin, W. Seok, and C. Shen, "Cellular SCTP: a transport-layer approach to Internet mobility," in *Proc. IEEE Intl. Conf. Computer Communications and Networks*, 2003, pp. 285–290.

[12] S. Fu, L. Ma, M. Atiquzzaman, and Y. Lee, "Architecture and performance of SIGMA: A seamless handover scheme for data networks," in *Proc. IEEE Intl. Conf. on Communications*, vol. 5, 2005, pp. 16–20.

[13] A. Ezzouhairi, A. Quintero, and S. Pierre, "A new SCTP mobility scheme supporting vertical handover," in *Proc. IEEE Intl. Conf. on Wireless and Mobile Computing, Networking and Communications*, 2006, pp. 205–211.

[14] W. Stevens, "RFC 2001: TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," *Request for Comments, IETF*, 1997.

[15] X. Yan, Y. Ahmet Sekercioglu, and S. Narayanan, "A survey of vertical handover decision algorithms in Fourth Generation heterogeneous wireless networks," *Comput. Netw.*, vol. In Press, Corrected Proof, 2010.

[16] A. Sgora and D. Vergados, "Handoff prioritization and decision schemes in wireless cellular networks: a survey," *IEEE Commun. Surv. Tutorials*, vol. 11, no. 4, pp. 57–77, 2009.

[17] J. Fitzpatrick, S. Murphy, M. Atiquzzaman, and J. Murphy, "Using cross-layer metrics to improve the performance of end-to-end handover mechanisms," *Comput. Commun.*, vol. 32, no. 15, pp. 1600–1612, 2009.

[18] I. Rec, "G. 107-The E Model, a computational model for use in transmission planning," *International Telecommunication Union*, 2003.

[19] L. Ma, F. Yu, and V. Leung, "Performance improvements of mobile SCTP in integrated heterogeneous wireless networks," *IEEE Trans. Wireless Commun.*, vol. 6, no. 10, pp. 3567–3577, 2007.

[20] S. Kashihara, K. Iida, H. Koga, Y. Kadobayashi, and S. Yamaguchi, "Multi-path transmission algorithm for end-to-end seamless handover across heterogeneous wireless access networks," in *Proc. Intl. Workshop on Distributed Computing*, 2003, pp. 836–836.

[21] S. Koh, M. Chang, and M. Lee, "mSCTP for soft handover in transport layer," *IEEE Commun. Lett.*, vol. 8, no. 3, pp. 189–191, 2004.

[22] E. Ribeiro and V. Leung, "Asymmetric path delay optimization in mobile multi-homed SCTP multimedia transport," in *Proc. ACM Workshop on Wireless Multimedia Networking and Performance Modeling*, 2005, pp. 70–75.

[23] L. Budzisz, R. Ferrús, A. Brunstrom, K. Grinnemo, R. Fracchia, G. Galante, and F. Casadevall, "Towards transport-layer mobility: evolution of SCTP multihoming," *Comput. Commun.*, vol. 31, no. 5, pp. 980–998, 2008.

[24] R. Ludwig and R. Katz, "The Eifel algorithm: Making TCP robust against spurious retransmissions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 1, pp. 30–36, 2000.

[25] E. Blanton and M. Allman, "RFC 3708: Using TCP duplicate selective acknowledgement (DSACKs) and stream control transmission protocol (SCTP) duplicate transmission sequence numbers (TSNs) to detect spurious retransmissions," *Request for Comments, IETF*, 2004.

[26] S. Ladha, S. Baucke, R. Ludwig, and P. Amer, "On making SCTP robust to spurious retransmissions," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 123–135, 2004.

[27] K. Zheng, M. Liu, Z. Li, and G. Xu, "SHOP: An integrated scheme for SCTP handover optimization in multihomed environments," in *Proc. IEEE Global Communication Conf.*, 2008, pp. 1–5.

[28] D. Kim and S. Koh, "Performance enhancement of mSCTP for vertical handover across heterogeneous wireless networks," *Int. J. Commun. Syst.*, vol. 22, no. 12, pp. 1573–1591, 2009.

[29] J. R. Iyengar, P. D. Amer, and R. Stewart, "Concurrent multipath transfer using sctp multihoming over independent end-to-end paths," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 951–964, 2006.

[30] G. Ye, T. Saadawi, and M. Lee, "IPCC-SCTP: An enhancement to the standard SCTP to support multi-homing efficiently," in *Proc. IEEE Intl.*

Conf. on Performance, Computing, and Communications, 2004, pp. 523–530.

[31] A. El Al, T. Saadawi, and L. M., "LS-SCTP: A bandwidth aggregation technique for stream control transmission protocol," *Comput. Commun.*, vol. 27, no. 10, pp. 1012–1024, 2004.

[32] J. R. Iyengar, P. D. Amer, and R. Stewart, "Performance implications of a bounded receive buffer in concurrent multipath transfer," *Comput. Commun.*, vol. 30, no. 4, pp. 818–829, 2007.

[33] J. Iyengar, P. Amer, and R. Stewart, "Retransmission policies for concurrent multipath transfer using SCTP multihoming," in *Proc. IEEE Intl. Conf. on Networks*, vol. 2, 2004, pp. 713–719.

[34] C. Casetti, W. Gaiotto, and D. di Elettronica, "Westwood SCTP: Load balancing over multipaths using bandwidth-aware source scheduling," in *Proc. IEEE Vehicular Technology Conf.*, vol. 4, 2004, pp. 3025–3029.

[35] P. Natarajan, N. Ekiz, P. Amer, and R. Stewart, "Concurrent multipath transfer during path failure," *Comput. Commun.*, vol. 32, no. 15, pp. 1577–1587, 2009.

[36] V. Bui, W. Zhu, A. Botta, and A. Pescapé, "A markovian approach to multi-path data transfer in overlay networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. In Press, Corrected Proof, 2010.

[37] E. Yilmaz, N. Ekiz, P. Natarajan, P. Amer, J. Leighton, F. Baker, and R. Stewart, "Throughput analysis of non-renegable selective acknowledgments (NR-SACKs) for SCTP," *Comput. Commun.*, vol. 33, no. 16, pp. 1982–1991, 2010.

[38] S. Mascolo, L. Grieco, R. Ferorelli, P. Camarda, and G. Piscitelli, "Performance evaluation of Westwood+ TCP congestion control," *Perform. Evaluation*, vol. 55, no. 1-2, pp. 93–111, 2004.

[39] R. Prasad, C. Dovrolis, M. Murray, and K. Claffy, "Bandwidth estimation: metrics, measurement techniques, and tools," *IEEE Network*, vol. 17, no. 6, pp. 27–35, 2003.

[40] R. Fracchia, C. Casetti, C. Chiasserini, and M. Meo, "WiSE: Best-path selection in wireless multihoming environments," *IEEE Trans. Mob. Comput.*, vol. 6, no. 10, pp. 1130–1141, 2007.

[41] K. Pentikousis, "TCP in wired-cum-wireless environments," *IEEE Commun. Surv.*, vol. 3, no. 4, pp. 2–14, 2000.

[42] B. Sardar and D. Saha, "A survey of TCP enhancements for last-hop wireless networks," *IEEE Commun. Surv. Tutorials*, vol. 8, no. 3, pp. 20–34, 2006.

[43] K. Leung and V. Li, "Transmission control protocol (TCP) in wireless networks: issues, approaches, and challenges," *IEEE Commun. Surv. Tutorials*, vol. 8, no. 4, pp. 64–79, 2006.

[44] S. Liu, S. Yang, and W. Sun, "Collaborative SCTP: A collaborative approach to improve the performance of SCTP over wired-cum-wireless networks," in *Proc. IEEE Intl. Conf. on Local Computer Networks*, 2004, pp. 276–283.

[45] S. Floyd, "TCP and explicit congestion notification," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 5, p. 23, 1994.

[46] G. Ye, T. Saadawi, and M. Lee, "Improving stream control transmission protocol performance over lossy links," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 4, 2004.

[47] E. Ribeiro and V. Leung, "Minimum delay path selection in multi-homed systems with path asymmetry," *IEEE Commun. Lett.*, vol. 10, no. 3, pp. 135–137, 2006.

[48] A. Argyriou and V. Madisetti, "Bandwidth aggregation with SCTP," in *Proc. IEEE Global Communication Conf.*, vol. 7, 2003.

[49] W. Xing, H. Karl, A. Wolisz, and H. Müller, "M-SCTP: Design and prototypical implementation of an end-to-end mobility concept," in *Proc. Intl. Workshop on The Internet Challenge: Technology and Applications*, 2002.

[50] A. Kelly, G. Muntean, P. Perry, and J. Murphy, "Delay-centric handover in SCTP over WLAN," *Autom. Control Comput. Sci.*, vol. 49, no. 63, pp. 1–6, 2004.

[51] L. Bokor, Á. Huszák, and G. Jeney, "On SCTP multihoming performance in native IPv6 UMTS-WLAN environments," in *Proc. ICST Intl. Conf. Testbeds and Research Infrastructures for the Development of Networks and Communities*, 2009, pp. 1–10.

[52] M. Chiu and M. Bassiouni, "Predictive schemes for handoff privatization in cellular networks based on mobile positioning," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 510–522, 2000.

[53] D. Lee and Y. Hsueh, "Bandwidth-reservation scheme based on road information for next-generation cellular networks," *IEEE Trans. Veh. Technol.*, vol. 53, no. 1, pp. 243–252, 2004.

[54] N. Samaan and A. Karmouch, "A mobility prediction architecture based on contextual knowledge and spatial conceptual maps," *IEEE Trans. Mob. Comput.*, vol. 4, no. 6, pp. 537–551, 2005.

[55] W. Soh and H. Kim, "A predictive bandwidth reservation scheme using mobile positioning and road topology information," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, p. 1091, 2006.

[56] T. Wallace and A. Shami, "An intelligent scheduling algorithm for concurrent multipath transfer using the stream control transmission protocol," *Submitted for Publication.*

Daniel Wallace received his B.E. degree in Software Engineering from Lakehead University, Thunder Bay, ON, Canada, in 2004, and his M.E.Sc. from the Univesrity of Western Ontario, ON, Canada, in 2007. He is currently pursuing a Ph.D. in Engineering Science at the University of Western Ontario. His research interests include grid computing, optical networks, and transport layer protocols.

Abdallah Shami received the B.E. degree in Electrical and Computer Engineering from the Lebanese University, Beirut, Lebanon in 1997, and the Ph.D. Degree in Electrical Engineering from the Graduate School and University Center, City University of New York, New York, NY in September 2002. In September 2002, he joined the Department of Electrical Engineering at Lakehead University, Thunder Bay, ON, Canada as an Assistant Professor. Since July 2004, he has been with The University of Western Ontario, London, ON, Canada where he is currently an Associate Professor in the Department of Electrical and Computer Engineering. His current research interests are in the area of wireless/optical networking.