Using Concurrent Multipath Transfer to Improve the SCTP Startup Behavior for PSTN Signaling Traffic

Karl-Johan Grinnemo and Anna Brunstrom
Department of Computer Science
Karlstad University, Karlstad, SWEDEN
Email: karlgrin,annab@kau.se
Jun Cheng
Ericsson AB, Stockholm, SWEDEN
Email: jun.dr.cheng@ericsson.com

Abstract—Although latency in the Internet has gained much attention in the research community, the latency issues of mobile control signaling have received less attention, and this all the while many telecom operators are experiencing a several-hundred percent increase in signaling traffic over only a couple of years. We believe one way to address both the latency and increased signaling load of mobile networks, is to exploit concurrent transfer of signaling traffic over several paths a.k.a. concurrent multipath transfer. This paper studies whether or not SCTP extended with concurrent multipath transfer (CMT-SCTP) could provide a faster startup behavior than standard SCTP. The paper complements previous work on CMT-SCTP, and extends it to PSTN signaling traffic. The paper suggests that CMT-SCTP could give a faster startup behavior over paths with similar bandwidths and round-trip times, but that its behavior is sensitive to differences in round-trip time between the paths. Moreover, the paper suggests that provided CMT-SCTP is configured with large enough send and receive buffers, it could provide a faster startup behavior than standard SCTP over a multipath association, in spite of some of the paths having a packet-loss rate of several percent.

Keywords—sctp; cmt-sctp; multipath; pstn; signaling; startup

I. INTRODUCTION

The current surge of interest in delay-sensitive applications, such as voice-over-IP and online gaming, have made latency one of the key challenges facing IP networks and the Internet. This is not least seen through high-profile efforts such as the ISOC workshop on reducing latency in the Internet [1], the bufferbloat initiative [2], and the EU FP7 project “Reducing Internet Transport Latency” (RITE) [3].

One very important delay-sensitive application area whose latency issues have to a large degree become eclipsed by the latency challenges of the Internet is the signaling in mobile networks. Signaling manages user data sessions, and, in smartphone-centric networks places a heavy demand on the mobile network control signaling plane – both at the radio interface and in the mobile core [4]. The source of this drastic increase in signaling traffic is to a large degree the usage patterns inherent with people’s mobile-connected lifestyles: mobile applications are designed to keep people connected throughout the day, and persistently polls for updates at different times, something that generates lots of uncoordinated connection requests and signaling.

The Stream Control Transmission Protocol (SCTP) [5] plays a key role in current and future Long-Term Evolution (LTE) mobile networks: It is employed in the transportation of signaling traffic between the evolved NodeB (eNB) in the Radio Access Network (RAN) and the Evolved Packet Core (EPC), i.e., the signaling and data traffic aggregation network of LTE, as well as within the IP Multimedia Subsystem (IMS), i.e., the subsystem within LTE responsible for multimedia services. And, as a core protocol of IMS, it is essential for the ongoing launch of the IMS-based services, Voice over LTE (VoLTE) and Rich Communication Services (RCS). We think one promising way to manage the increasing signaling traffic loads and latencies in current and future mobile networks is to explore low-latency communication over multiple paths in SCTP.

Originally, multi-homing in SCTP was built in to support path-level redundancy: The equivalent to a link in a circuit-switched telephony network is a network path in an IP network. In order to provide the same level of fault resilience as the one offered in a circuit-switched network, multi-homing was included in SCTP. However, there was no support for concurrent multipath transfer: User messages were only sent over one path at a time, the primary path; the remaining alternate paths only served as backup paths in case the primary path became unreachable.

In the past several years, concurrent multipath transfer, has received a lot of attention in the research community, not least in the IETF. At the time of writing, there is ongoing work in IETF to standardize concurrent multipath extensions for both TCP (Multipath TCP) [6] and SCTP (CMT-SCTP) [7]. The standardization of CMT-SCTP is primarily made on the basis of the work of Iyengar et al. [8]. They identified and resolved the negative side effects of packet reordering introduced by concurrent multipath transfer, and also made feasible through simulations and experiments that both the startup and stationary behavior of SCTP could be improved by CMT. This paper complements their work by extending it to telephony (PSTN) signaling traffic. The paper compares the startup behavior of CMT-SCTP with that of standard SCTP, i.e., SCTP as specified in RFC 4960 [5]. It considers the startup behavior over symmetrical as well as asymmetrical paths, i.e., paths with similar and dissimilar round-trip times (RTTs),
bandwidths, and/or packet loss. Moreover, the paper considers the impact of signaling traffic characteristics, notably, message sizes, intensity and burstiness, on the standard SCTP and CMT-SCTP startup behaviors. One reason why CMT-SCTP could provide a faster startup than standard SCTP is that it could provide a faster startup than standard SCTP is that it under ideal conditions is able to send \( n \) times as many packets per transmission round, with \( n \) being the number of paths in an association: During slow start, CMT-SCTP opens up \( n \) congestion windows in parallel, not only one. The paper corroborates this advantageous property of CMT-SCTP.

The remainder of the paper is organized as follows. Section II provides a description of the experiment setup and methodology. The outcome of the experiment is covered by Sections III to V. Section VI summarizes prior and contemporary related work. The paper concludes in Section VII.

### II. Experiment Setup

To compare the startup behavior of CMT-SCTP with that of standard SCTP, we used the testbed depicted in Figure 1. The studied flow was sent from Host S to Host R and was generated by a traffic generator developed by us, \( \text{tgen} \). The \( \text{tgen} \) traffic generator generated reliable ordered traffic with customizable characteristics over a single stream. In the tests with standard SCTP, the studied flow was sent on path \( P_1 \), and in the tests with CMT-SCTP, on both paths \( P_1 \) and \( P_2 \). Since PSTN signaling traffic is typically transported over a managed IP network with redundant network paths between signaling endpoints, we deliberately did not consider scenarios in which the paths in a CMT-SCTP association have a shared bottleneck link. The two dual-homed hosts, Host S and Host R, were running FreeBSD 9.0, and utilized the FreeBSD implementation of SCTP [9] and its extension for concurrent multipath transfer [8].

The experiment comprised the tests listed in Table I. The studied traffic flow was generated in bursts with an inter-burst interval uniformly distributed between [1,4] ms. The burst length was set so that the traffic flow had a certain average send rate. As follows from the table, three average send rates were considered: 7 Mbps, 21 Mbps and "nl". A send rate of "nl" denotes a network-limited send rate, i.e., packets were sent as fast as the network path permitted. Also, the message size of the studied traffic flow was varied. Both the send rates and the message sizes were selected to mimic the characteristics of packet-based PSTN signaling traffic [10], and have been used in previous PSTN signaling traffic studies by Ericsson.

The RTTs were selected to cover a broad spectrum of path delays and delay asymmetries, and the RTTs above 50 ms should be seen as extreme values rather than be taken as typical delays in LTE networks. To evaluate the effects of packet losses on the startup behavior of standard SCTP and CMT-SCTP, respectively, we ran the symmetrical-path tests with five packet-loss rates: 1%, 3%, 5%, 10% and 15%. The packet-loss rates were selected to cover light, moderate, and high packet-loss conditions [11]. Again, as for the RTTs, the higher values should be seen as extreme values. To limit the number of tests, those packet-loss-rate settings that are suffixed with a dagger have only been run with send rates of 7 Mbps and 21 Mbps, and with a message size of 100 Bytes. Sample test runs suggested that there are no significant difference in the outcome when the send rate is increased from 21 Mbps to "nl". Also the message size was shown to have a limited impact on the startup behavior (cf. Section III).

The FreeBSD kernel settings are provided in Table II. The FreeBSD kernel settings are provided in Table II. The FreeBSD kernel settings are provided in Table II. Since coupled congestion control for CMT-SCTP has not yet been standardized, tests were only run with "plain" CMT-SCTP, i.e., with coupled congestion control disabled. Also, it could be noted that CMT-SCTP employs the so-called RTX-STHRESH retransmission policy [8], i.e., retransmissions are made to the destination for which the sender has the largest slow-start threshold. Both the send and receive buffers were configured to avoid socket buffer blocking, and buffer

---

### Table I: Test Settings

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTT ([P_1, P_2] ) (ms)</td>
<td>[10.10], [10.50], [10.210], [40.40], [40.140], [40.240], [100.100]</td>
</tr>
<tr>
<td>Bandwidth ([P_1, P_2] ) (Mbps)</td>
<td>[5.42], [42.42]</td>
</tr>
<tr>
<td>Packet-loss rate ([P_1, P_2] ) (%)</td>
<td>[0.0], [0.1], [0.3], [0.10], [1.00], [1.11], [1.21], [1.31], [1.51], [2.11], [2.21], [2.31], [2.51], [3.11], [3.21], [3.31], [5.00], [5.11], [5.21], [5.51], [10.00]</td>
</tr>
<tr>
<td>Average Send rate (Mbps)</td>
<td>7, 21, nl</td>
</tr>
<tr>
<td>Message Size (Bytes)</td>
<td>50, 100, 400</td>
</tr>
</tbody>
</table>

---

### Table II: FreeBSD Kernel Settings

<table>
<thead>
<tr>
<th>Kernel Parameter</th>
<th>SCTP</th>
<th>CMT-SCTP</th>
</tr>
</thead>
<tbody>
<tr>
<td>kern.ipc.mbufclusters</td>
<td>15346</td>
<td>15346</td>
</tr>
<tr>
<td>kern.ipc.maxsockbuf</td>
<td>20 MiBytes</td>
<td>20 MiBytes</td>
</tr>
<tr>
<td>net.inet.sctp.cmt_on_off</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>net.inet.sctp.cmt_turned_off</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>net.inet.sctp.buffer_splitting</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net.inet.sctp.ns sacked_on_off</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>net.inet.sctp.sendpace</td>
<td>1/4 MiBytes</td>
<td>1/4 MiBytes</td>
</tr>
<tr>
<td>net.inet.sctp.recvpace</td>
<td>7 MiBytes</td>
<td>7 MiBytes</td>
</tr>
<tr>
<td>net.inet.sctp.rto_initial</td>
<td>100 ms</td>
<td>100 ms</td>
</tr>
<tr>
<td>net.inet.sctp.rto_min</td>
<td>900 ms</td>
<td>900 ms</td>
</tr>
<tr>
<td>net.inet.sctp.sto_min</td>
<td>10 ms</td>
<td>10 ms</td>
</tr>
<tr>
<td>net.inet.sctp.delayed_sack_time</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>net.inet.sctp.initial_cwnd</td>
<td>1 and 6 MTUs</td>
<td>1 and 6 MTUs</td>
</tr>
</tbody>
</table>

---

![Figure 1: Testbed used in our comparison of the startup behavior of CMT-SCTP with that of standard SCTP.](image-url)
Fig. 2. Comparison of startup behavior between CMT-SCTP and standard SCTP over symmetrical paths. The studied traffic was sent at 21 Mbps.

(a) Traffic with a message size of 50 Bytes.

(b) Traffic with a message size of 400 bytes.

Fig. 3. Comparison of startup behavior between CMT-SCTP and standard SCTP over asymmetrical paths. The studied traffic was sent at 21 Mbps.

(a) The paths have the same RTT but differ in bandwidth.

(b) The paths have same bandwidth but differ in RTT.

III. STARTUP OVER LOSS FREE PATHS

In the symmetrical-path tests both paths had a bandwidth of 42 Mbps; three RTTs were considered, 10 ms, 40 ms, and 100 ms, and there were no explicit packet losses. Figure 2 plots the cumulative received data for two of the 21-Mbps test series: two test series whose outcome are fairly representative for the outcome of all the symmetrical-path tests. As follows, CMT-SCTP had on the average a faster startup behavior in both these test series. In fact, CMT-SCTP gave a better result than standard SCTP in almost all symmetrical-path tests. Moreover, since a longer RTT increased the lengths of the transmission rounds, the gain in startup for CMT-SCTP increased with longer RTTs.

As regards the asymmetrical-path tests, the bandwidth of $P_2$ was fixed to 42 Mbps while the bandwidth of path $P_1$ took on the values 5 Mbps and 42 Mbps, respectively. Again, there were no explicit packet losses. Seven different combinations of RTTs on the two paths were studied, covering both long and short path delays, and path-delay differences.

1 Buffer splitting means that the send and/or receiver buffer is split into separate parts — one part for each path — and has been introduced to prevent some paths from allocating too much buffer space, and in so doing inhibit transfer on other paths.
Two significative series of asymmetric-path tests are shown in Figure 3. Figure 3(a) illustrates the effect of bandwidth asymmetry. We observe that the average startup behavior of CMT-SCTP was faster than for standard SCTP in these test series. This actually held true in more or less all tests with bandwidth asymmetry. As expected, CMT-SCTP was able to exploit the extra bandwidth of path P₂ without introducing more packet reordering than the send and receive buffers were able to handle. The reason standard SCTP with an RTT of 40 ms sometime into the test runs had a worse evolution of the cumulative received data than standard SCTP with an RTT of 100 ms, was because of an unfortunate packet loss that had to be recovered through a timeout and retransmission on the alternate path. In the corresponding test runs with standard SCTP and an RTT of 100 ms, there were no timeouts.

Figure 3(b) illustrates the effect of RTT asymmetry, i.e., when the two paths had different RTTs but the same bandwidth. In these test series, standard SCTP outperformed CMT-SCTP, and this was also the case for the rest of the studied test series with RTT asymmetry. The “stair-case” shape of the graphs in the figure for CMT-SCTP is an effect of head-of-line blocking: packets sent on path P₁ were being delayed by packets sent on P₂ since path P₁ had a shorter RTT than path P₂. Probably, we would have observed cases in which CMT-SCTP improved on standard SCTP, if we had also considered scenarios where path P₂ had a shorter RTT than P₁. It should also be noted that there are several proposals on how to properly schedule packets over several paths in an SCTP association to maximize the chances of ordered delivery, which could be employed by CMT-SCTP to mitigate the effects of head-of-line blocking [12], however, none of these proposals have been considered mature enough for standardization.

When it comes to the tests with both RTT and bandwidth asymmetry, CMT-SCTP seemed to perform better than standard SCTP when the difference in RTT between the two paths was small, and conversely, a large difference in RTT between the paths seemed to benefit standard SCTP. As to the traffic characteristics, the outcome of the tests suggested that there was no clear correlation between the difference in startup behavior of CMT-SCTP and standard SCTP in terms of burstiness and message size.

IV. IMPACT OF PACKET LOSS
As already mentioned, five packet-loss rates were considered: 1%, 2%, 3%, 5%, and 10%; the packet losses were uniformly distributed. Both symmetrical as well as asymmetrical packet-loss scenarios were considered, i.e., tests were conducted with both equal as well as unequal packet-loss rates on paths P₁ and P₂.
we observed an increased gain in faster startup with CMT-SCTP packet-loss rate on path $P_1$ of 10 ms and 100 ms, respectively.

rate of 21 Mbps, a message size of 100 Bytes, and for R TTs
outcome from a representative select of the tests with a send and R TT. Figure 4 illustrates our findings by showing the
with standard SCTP increased with increased packet-loss rate and R TT.

Furthermore, as before, we observed an increased gain in faster startup with CMT-
SCTP with an increased RTT. The reason CMT-SCTP could, despite packet losses, still benefit from transmitting packets on path $P_2$ was probably due to it being configured with large send and receive buffers: The send and receive buffers had to be large enough to accommodate out-of-order and dropped packets in all tests; especially in the tests with the largest sum of path bandwidth-delay products, i.e., the tests with an RTT of 40 ms on path $P_1$, an RTT of 240 ms on $P_2$, and with a bandwidth of 40 Mbps on both paths. As follows from Dreibholz et al. [13], the sender and receiver buffers had to be around 5 MiBytes, something which was more than met by our CMT-SCTP configuration (cf. Table II). The outcome of the asymmetrical tests with higher packet-loss rate on path $P_1$ than $P_2$ was as to be expected: CMT-SCTP gave a faster startup than standard SCTP in all considered tests. Again, we noted that the difference in startup performance between CMT-SCTP and standard SCTP increased with increased RTT. Figure 6 exemplifies our findings.

Fig. 7. Comparison between CMT-SCTP with a default IW (3 MSS), and standard SCTP with IW set to twice the size of the default setting (6 MSS). The studied traffic was sent at 21 Mbps with a message size of 100 Bytes.

To begin with, let us consider the tests with symmetrical packet-loss rates. CMT-SCTP exhibited a faster startup behavior than standard SCTP. Notably, in all 21-Mbps tests, irrespective of RTT and packet-loss rate, CMT-SCTP exhibited a faster startup behavior than standard SCTP. Furthermore, in all tests with symmetrical packet-loss rates, we observed that the gain in faster startup of using CMT-SCTP as compared with standard SCTP increased with increased packet-loss rate and RTT. Figure 4 illustrates our findings by showing the outcome from a representative select of the tests with a send rate of 21 Mbps, a message size of 100 Bytes, and for RTTs of 10 ms and 100 ms, respectively.

Next, let us consider the asymmetrical tests with higher packet-loss rate on path $P_2$, the alternate path of SCTP, than on path $P_1$, the primary path of SCTP. Figure 5 provides the result for the tests with the same send rate, message size, and RTT as in Figure 4. One might have expected that buffer blocking would have occurred and impeded on the CMT-SCTP startup, however, this was not the case. Instead, CMT-SCTP provided a faster startup in the majority of these tests as well. In fact, standard SCTP only gave a faster startup in a few 7-Mbps tests in which the RTT was 10 ms, and the packet-loss rate was 1% on path $P_1$. Furthermore, as before, we observed an increased gain in faster startup with CMT-

V. CMT VS. INCREASED INITIAL WINDOW

There are primarily two reasons why CMT-SCTP in certain cases gives a faster startup compared to standard SCTP. The first and most obvious one being that CMT-SCTP is able to exploit the bandwidth of several network paths – not...
just one, as is the case for standard SCTP. Consider for example our tests in which we tried to transmit 7 Mbps traffic over an association consisting of a 5 Mbps (primary) and a 42 Mbps (alternate) path. In these tests, CMT-SCTP were able to accommodate this traffic by striping it over both paths, while SCTP had to throttle the traffic. The other reason being that CMT-SCTP ideally is able to send $n$ times as many packets per transmission round during slow start than standard SCTP, where $n$ is the number of paths in the association: Although CMT-SCTP and standard SCTP open up their congestion window at the same pace over a single path, CMT-SCTP is able to open up the congestion window of $n$ paths in parallel. To validate the latter, we re-ran the symmetrical tests with SCTP configured with an IW of twice its default size, i.e., 6 MSS instead of 3 MSS. The outcome of these tests was then compared with the corresponding CMT-SCTP tests. Figure 7 plots the result from the same select of tests as was plotted in Figure 2. As expected, the startup behavior of CMT-SCTP was roughly the same as for standard SCTP with twice the initial congestion window, and this observation was not only true for these tests, but also for the rest of the studied symmetrical-path tests. Consequently, an association with $n$ paths in CMT-SCTP shortens the startup delay as much as standard SCTP with an IW of $n$ times its default size, however, does so without introducing more congestion and queuing delays on these paths than a standard SCTP flow would do.

VI. Related Work

As shown in Budzisz et al. [14], there are a number of works on extending standard SCTP with concurrent multipath transfer. One of the initial works on concurrent multipath transfer for SCTP, Load Sharing SCTP (LS-SCTP), was carried out by Al et al. [15], [16]. In order to separate the flow control from the congestion control, LS-STCP introduced an additional sequence number, a path sequence number, and a new SACK chunk for acknowledging chunks on a per-path basis. However, these additions constitute a major problem with LS-SCTP: It makes LS-SCTP incompatible with standard SCTP, and the per-path sequence numbers introduce extra overhead in the DATA chunks. In an attempt to address this, Ye et al. [17] proposed the Independent per-Path Congestion Control SCTP (IPCC-SCTP). IPCC-SCTP builds upon LS-SCTP and the idea of per-path congestion control, but requires less number of changes to standard SCTP than LS-SCTP does. Particularly, it only introduces one additional sequence number, a path sequence number, and does not impose any changes to the data chunk format.

More or less in parallel with the work of Al et al., Iyengar et al. [18] considered ways of extending SCTP with concurrent multipath transfer, without introducing per-path sequence numbers and/or new SACK chunks. Their research resulted in CMT-SCTP [8], which maintains per-destination virtual queues to keep track of data chunks and their paths. Retransmissions are only triggered when several SACKs report missing data chunks from the same virtual queue. A further improvement of CMT-SCTP was proposed by Jungmaier and Rathgeb [19]. They proposed path-based SACKs that sought to improve CMT-SCTP by maintaining a SACK counter for each path instead of only one per association. The rationale behind path-based SACKs was that a strict allocation of SACK chunks to their respective paths would be useful for the SACK clocking: two successive SCTP packets with data chunks arriving on different paths triggered two SACKs on these paths instead of just one. At the time of writing, a concurrent multipath extension to SCTP that builds upon CMT-SCTP is under consideration for standardization in IETF [7].

As mentioned in the introduction, simulations and experiments by Iyengar [8] suggest that both the startup and stationary behavior of standard SCTP could be improved by introducing their concurrent multipath extensions. However, in both their simulations and experiments they only considered bulk transfer: real-time traffic and timeliness aspects were not considered. Our work complements their’s by studying the startup performance of CMT-SCTP for real-time PSTN signaling traffic. And, by limiting us to the startup behavior of CMT-SCTP, we have been able to do a more comprehensive study than was feasible in their case. Moreover, our study also encompasses the effects of path asymmetry, something which was not within the scope of their work. Still, later simulations by Dreibholz et al. [13], [20] and Sarwar et al. [21] have indeed considered path asymmetries in terms of bandwidth and packet loss. Our asymmetric-path tests validate their simulations, and also give further insight on how the effects of path asymmetry vary with the characteristics of the traffic, notably PSTN signaling traffic.

VII. Conclusion

This paper compares the startup behavior of PSTN signaling flows over CMT-SCTP as compared with standard SCTP when the paths in an association have the same or different RTTs, similar or dissimilar bandwidths, and equal or unequal packet-loss rates. Further, it investigates the impact of the traffic burstiness and the sizes of the signaling messages on the startup behavior. The paper suggests that CMT-SCTP provides a faster startup behavior than standard SCTP over paths with similar RTTs and bandwidths, and that the gain increases with increasing RTT. In fact, provided there was enough available bandwidth, the startup behavior of CMT-SCTP was roughly the same as for standard SCTP with twice the initial congestion window. Furthermore, the paper indicates that the benefit from using CMT-SCTP remains in spite of some differences in bandwidth or packet loss between the paths. Particularly, the paper demonstrates that provided CMT-SCTP is appropriately configured with large send and receive buffers, it could provide a faster startup behavior than standard SCTP over a multipath association, although some of the paths in the association have a packet-loss rate of several percent. Finally, it could be worth mentioning that there was no clear correlation between the difference in startup behavior of CMT-SCTP and standard SCTP in terms of traffic burstiness and message size. In our future work, we intend to extend CMT-SCTP with solutions for low-latency communication. To avoid so-called head-of-line blocking, SCTP and CMT-SCTP support multiple independent streams. We intend to complement the CMT-SCTP stream and path management with low-latency stream
and path scheduling schemes, and with appropriate latency-aware congestion control. Furthermore, to reduce costs and simplify the operation and maintenance of the LTE EPC/IMS there are ongoing efforts within leading telecom equipment manufacturers, e.g., Ericsson and Huawei, to migrate these systems to the cloud. Our future work entails considering latency-aware CMT-SCTP solutions for an EPC/IMS cloud.

REFERENCES