TCP Extensions for Multipath Transport
- Based on IETF Working Group MPTCP

Amanpreet Singh

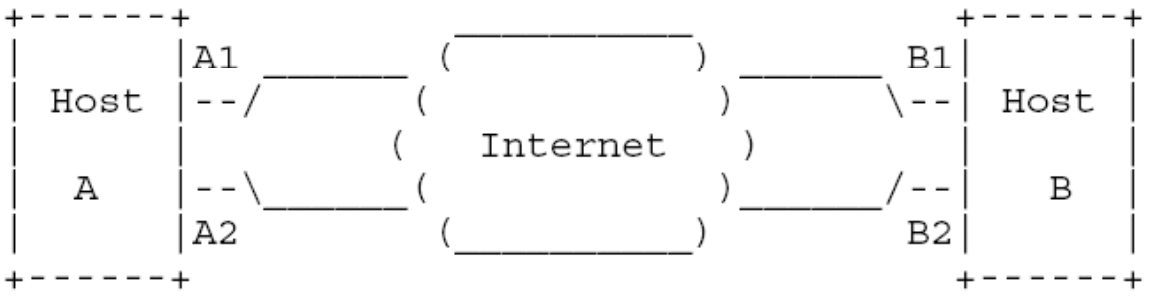
3rd September 2010

Outline

- Motivation
- Goals
- Compatibility Requirement
- Multipath TCP Architecture
- Multipath TCP Concept and Signaling
- Linked Congestion Control
- Multipath TCP Solutions (MPTCP, MCTCP, PLMT)
- Multipath Transport Implementation (Linux Kernel, QualNet Simulator)
- Conclusion & Outlook
- References

Motivation

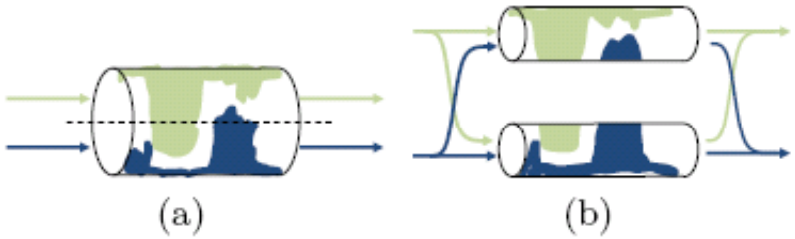
- Demands on Internet resources are ever-increasing, but often these resources (in particular, bandwidth) cannot be fully utilized due to protocol constraints both on the end-systems and within the network
  - TCP/IP communication is currently restricted to a single path per connection [1]
- In addition, there is an increasing trend for small mobile devices to have access to multiple technologies for connecting to the Internet
  - This gives researchers an increasing interest for solutions allowing to use efficiently several communication mediums.



- Thus an ongoing research for the Multipath TCP solution, that allows spreading of a single TCP flow across multiple Internet paths, without requiring any change to applications

Goals

In providing the use of multiple paths, multipath TCP has the following functional goals [3].



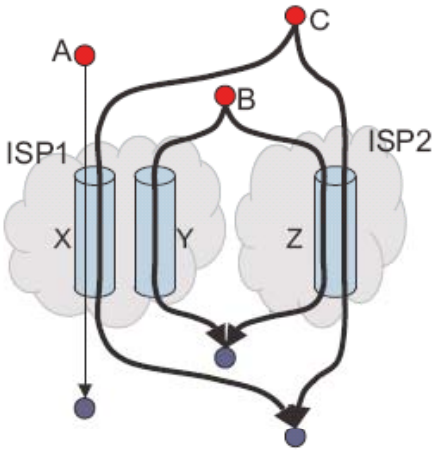
(a) balancing across dissimilar speed links

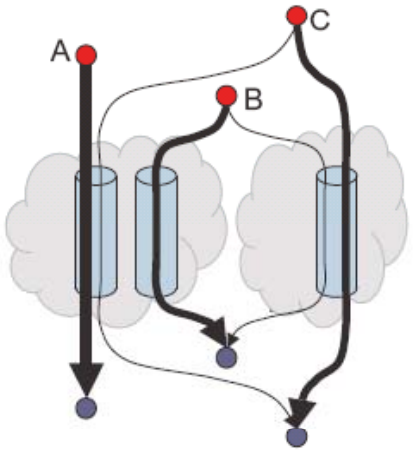
(b) balancing across dissimilar cost links

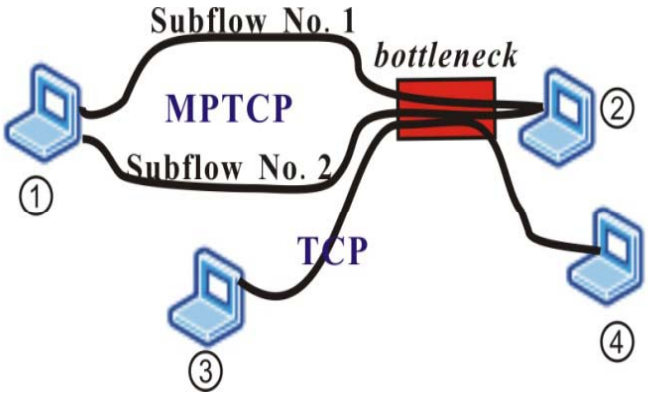
- **Improve Throughput**: Multipath TCP must support the concurrent use of multiple paths to increase the efficiency of the resource usage, and thus increase the network capacity available to end hosts
- **Improve Resilience**: Multipath TCP must support the use of multiple paths interchangeably , by permitting packets to be sent and re-sent on any available path
- **Balance congestion**: A multipath flow should move as much traffic as possible off its most congested paths, subject to meeting the first two goals

Compatibility Requirement

- **Application Compatibility**: Multipath TCP should be compatible with existing TCP APIs and follow the in-order, reliable and byte-oriented delivery service model of TCP [1]
- **Network Compatibility**: Multipath TCP should be compatible with the Internet as it exists today, able to traverse middleboxes such as firewalls, NATs, and performance enhancing proxies [4]
- **Compatibility with other Network Users**: Multipath TCP should do no harm to legacy connections (coexist gracefully with existing legacy TCP flows)

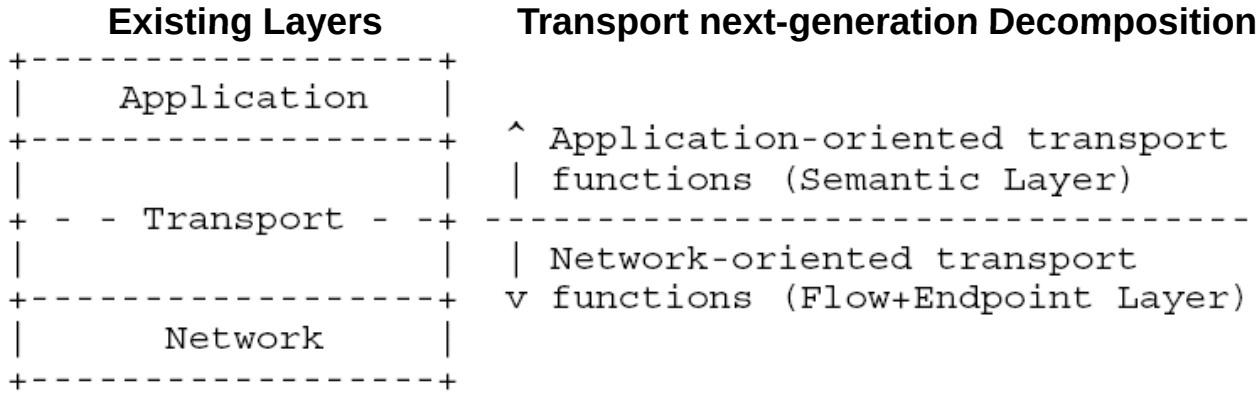




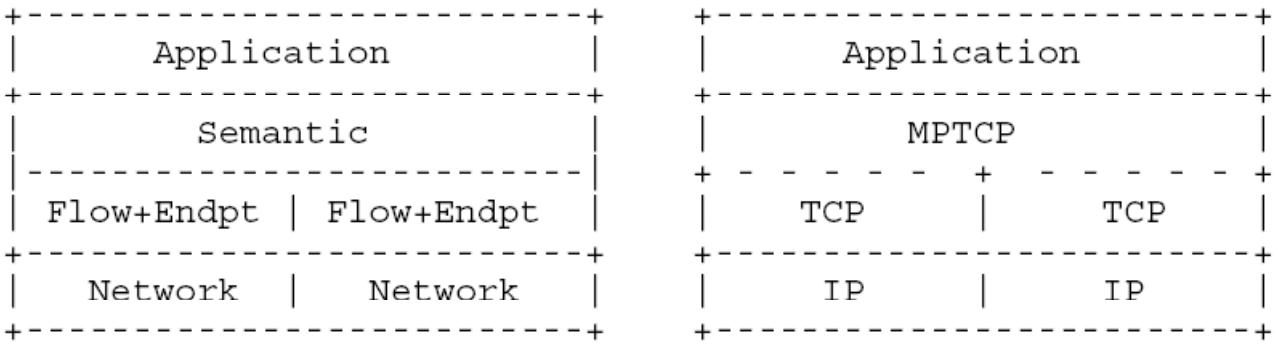


Multipath TCP Architecture

- The new Internet model described here is based on ideas proposed earlier in “Transport next-generation„ (Tng)[5].



Mapping of ‚Tng‘ Decomposition to MPTCP Architecture [3]



Multipath TCP Concept

- Multipath TCP (MPTCP) is a set of extensions for TCP that allows spreading of a single TCP flow across multiple Internet paths.
- Each subflow looks to the network as a normal TCP flow, with the only difference that it carries new multipath TCP signaling.
  - Multipath TCP signaling is used to declare MPTCP support, exchange alternate addresses and other control messages.
- The MPTCP proposal uses a dual sequence number space
  - Each subflow has its own sequence space that identifies bytes within a subflow as if it were running alone.
  - There is also a data (or connection level) sequence space, which allows reordering at the (aggregated) connection level.
  - Each segment carries both subflow and data sequence numbers.
  - Retransmissions are driven only by the subflow sequence number; hence MPTCP avoids problems due to connection level reordering of packets.
- In MPTCP, congestion control is coupled across paths, so as to ensure fairness without needing to detect shared bottlenecks.
- MPTCP performs flow control in aggregate (not on individual subflows).

Multipath TCP Signaling

MPTCP operations needs the following signaling information in the TCP segments, either in the TCP options field or payload

- **“Multipath Capable (MPCap)”:** informing the sender/receiver/middleboxes of the multipath capability of the endpoints and their intention to use it.
- **“Join Connection”:** for the setup of additional subflows, identify to which connection the additional subflow belongs to.
- **“Add Address”:** Exchange address information, notifying the other endpoint about the additional available interfaces at the sender of the information.
- **“Data Sequence Number”:** The connection-level data sequence number is added to keep in-order delivery over the multiple subflows.
- **“Data Acknowledgement”:** The data acknowledgment is utilized to record connection-level accumulative acknowledgement.
- **“Data FIN”:** The data FIN is used to terminate the whole MPTCP connection.

Linked Congestion Control for MPTCP

- MPTCP obtains an unfair share in the bottleneck when coexisting with TCP connections, if TCP congestion control algorithms are run separately on each sublow of a MPTCP connection.
- Linked congestion control [6][7] couples all the subflows of a Multipath TCP connection in order to control the aggressiveness of the subflows in a congested state:
  - In the **congestion avoidance state**, each time when a new ACK arrives for the subflow *i*, the corresponding subflow congestion window *cwnd_i* is increased by

$$\mathrm{MIN}(\mathrm{ceil}(\mathrm{alfa} * \mathrm{bytes\_acked} * \mathrm{mss\_i} / \mathrm{tot\_cwnd}), \mathrm{bytes\_acked} * \mathrm{mss\_i} / \mathrm{cwnd\_i})$$

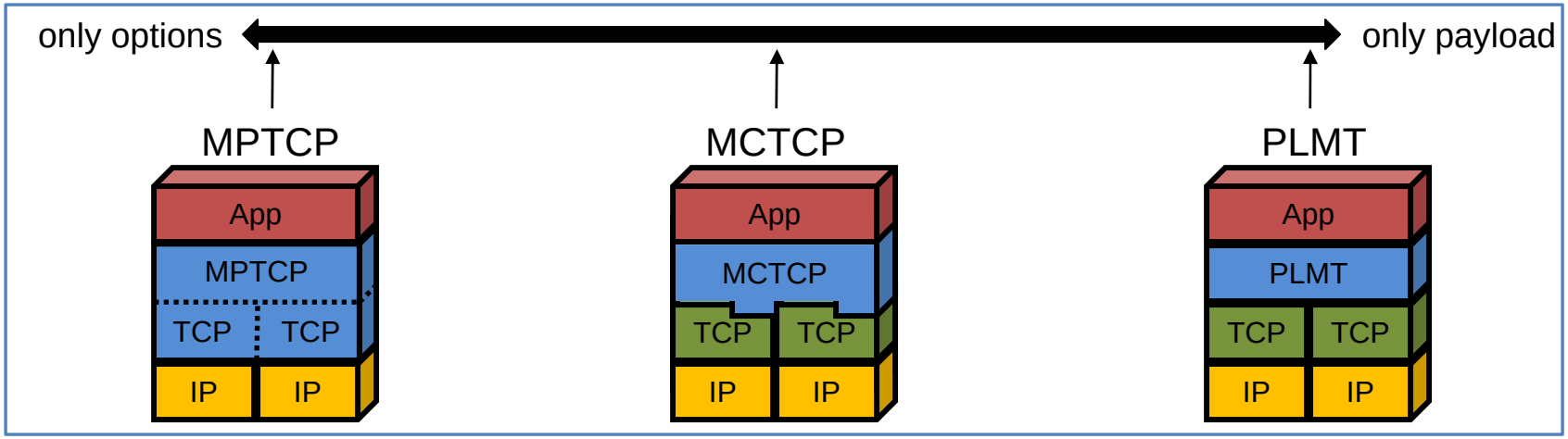
where,

$$\mathrm{alfa} = \mathrm{tot\_cwnd} * \frac{\max_i \frac{\mathrm{cwnd\_i} * \mathrm{mss\_i}^2}{\mathrm{rtt\_i}^2}}{\left\{\mathrm{sum}_i \frac{\mathrm{cwnd\_i} * \mathrm{mss\_i}}{\mathrm{rtt\_i}}\right\}^2}$$

  - In case of a packet loss on the subflow *i*, the congestion window *cwnd_i* is reduced based on the standard TCP New Reno congestion control mechanism

Multipath TCP Solutions

- Each multipath TCP (MPTCP)[8] subflow looks to the network as a normal TCP flow, with the only difference that it carries new TCP options for MPTCP signaling.
- Payload Multi-connection Transport (PLMT) [9] is a multipath protocol variant that encodes all the signaling information in the payload of TCP connections.
  - Operates as an additional protocol layer on top of existing TCP (no change in existing TCP)
  - Multipath initiation and control via separate *control connection* to a well-known port
  - Can be realized entirely in the user-space of an operating system
- Multi-Connection TCP (MCTCP) [10] transport is a hybrid variant that encodes control information, as far as possible, in the payload of the TCP connections
  - Initially uses the TCP option field in the connection setup messages (SYN/ACK for MPCap and Join)
  - It is transparent in the single-path case.



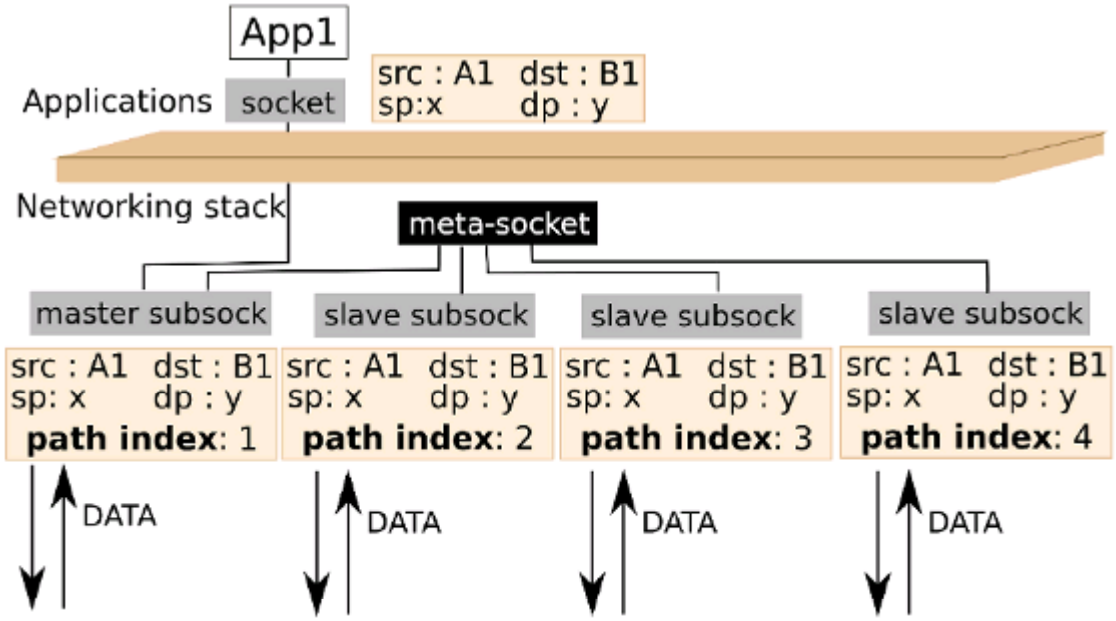
Multipath TCP Solutions: Comparison Table

Criterion		MPTCP	MCTCP	PLMT
Changes inside TCP/IP network stack		Significant	Small	None (in best case)
connection setup	Signaling	TCP options	TCP options	TCP payload (using control connection)
	Potential start-up delay (no troublesome middlebox present)	None	None	None** or about 2x RTT *
connection control		TCP options	TCP payload	TCP payload
Extensibility		Difficult due to limited options' space	Simple	Simple
Robustness to middleboxes	… preventing unknown options in SYNs	Fallback to legacy TCP after timeout	Fallback to legacy TCP after timeout	Not affected
	… preventing unknown options outside SYNs	Fallback to legacy TCP, potentially even failure	Not affected	Not affected
	… changing payload	Terminate affected sub flows, option to fall back to legacy TCP	Terminate affected sub flows, option to fall back to legacy TCP	Terminate affected sub-flows, currently no fall-back option
Control by middleboxes	Change of control information	Simple	Difficult	Simple if access to control connection is available, difficult in other cases
Server vulnerability to …	… breaking connections	Only MPTCP connections	Only MCTCP connections	All TCP connections

* assuming control connection is set up before data connection
** assuming control connection is set up after data connection

MPTCP Kernel-space Implementation

- Connection-specific information is held in a new structure at the connection-level, called meta-socket
- The master socket is a special socket as it is the only connection to the application

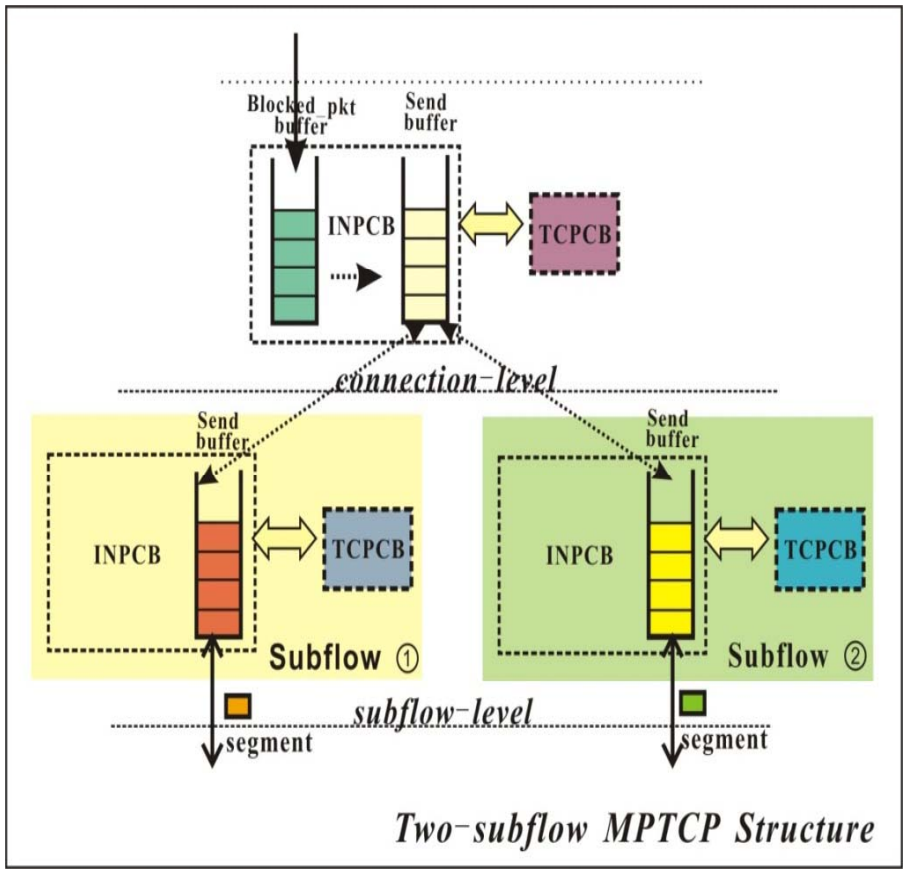


Linux MPTCP architecture [11]

- Data arriving on the subflows is serviced by the master and slave sockets (checking for in-order, in window sequence numbers, etc.), and passed to the meta-socket once it is in order at subflow level.

MPTCP Simulator Implementation: QualNet

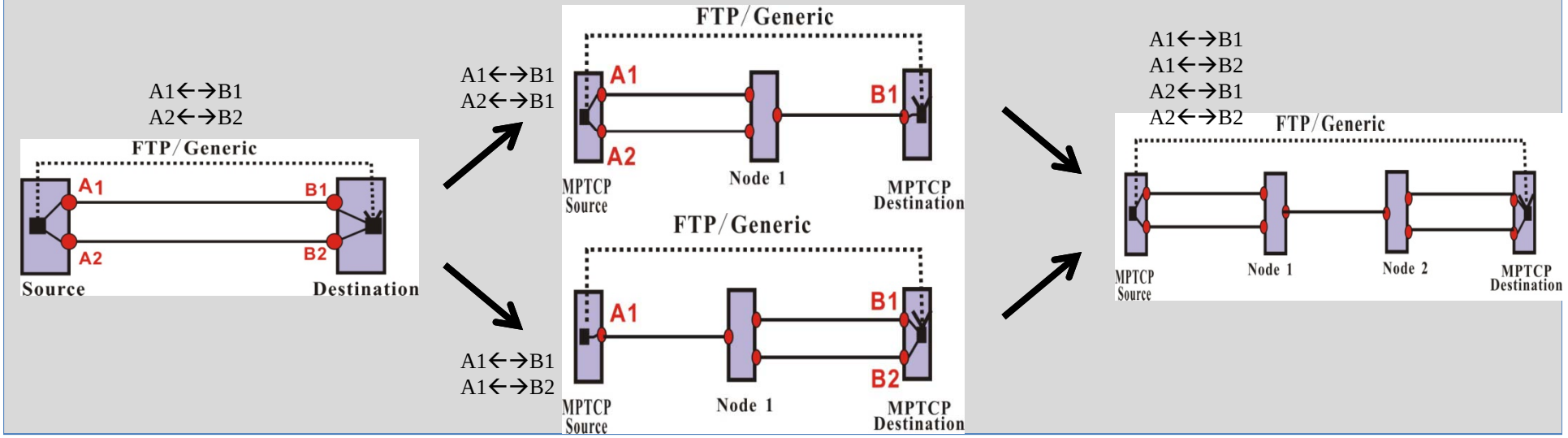
Architecture inside the Two-Subflow MPTCP Protocol



- There are two subflows linked to the same connection-level send buffer.
- Each subflow has its own address information. The address for the default subflow is equal to that of the connection level.
- The subflow INPCB instances of a connection are identified by the same *connection_id*.
- Congestion control is subflow specific.

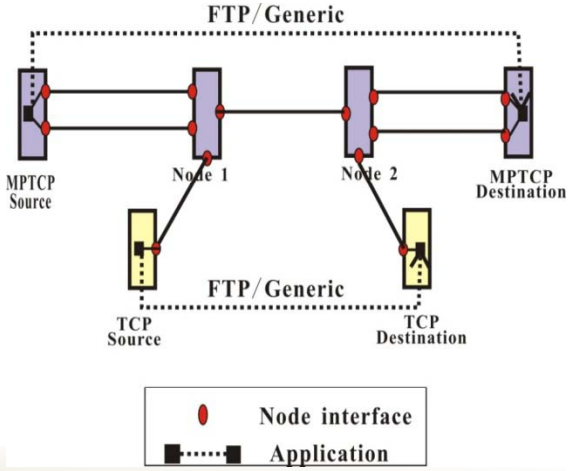
MPTCP Implementation Scenarios

1: Two-to-four-subflow setup with the MPTCP Protocol



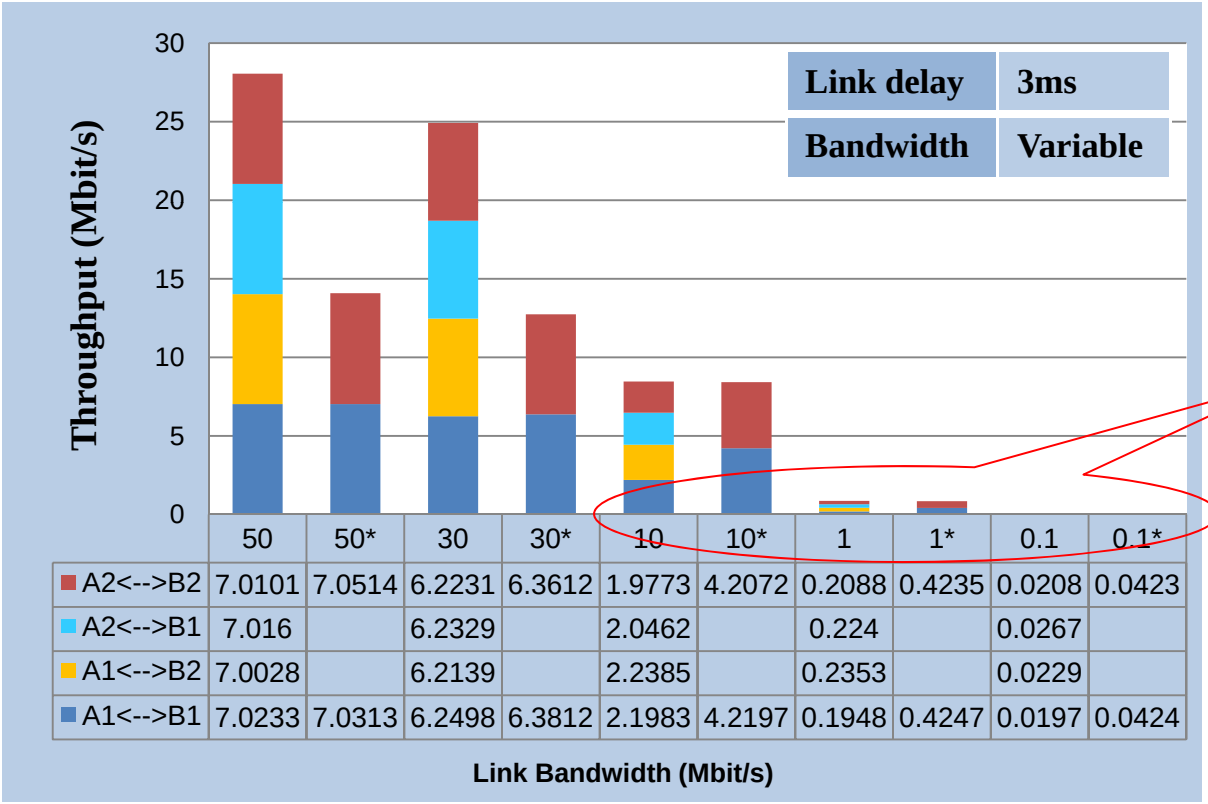
Simulation Parameters	Value
Simulation time	10000s
FTP Application process data length	5,120 Kbyte
TCP/Subflow send/receive buffer size	16,384 byte
MPTCP connection-level send/receive buffer size	163,840 byte
TCP/Subflow congestion control variant	NewReno

2: MPTCP and TCP connections coexist



Scenario 1: Performance Evaluation (1)

Comparison of Throughput for the two-subflow and four-subflow MPTCP connections

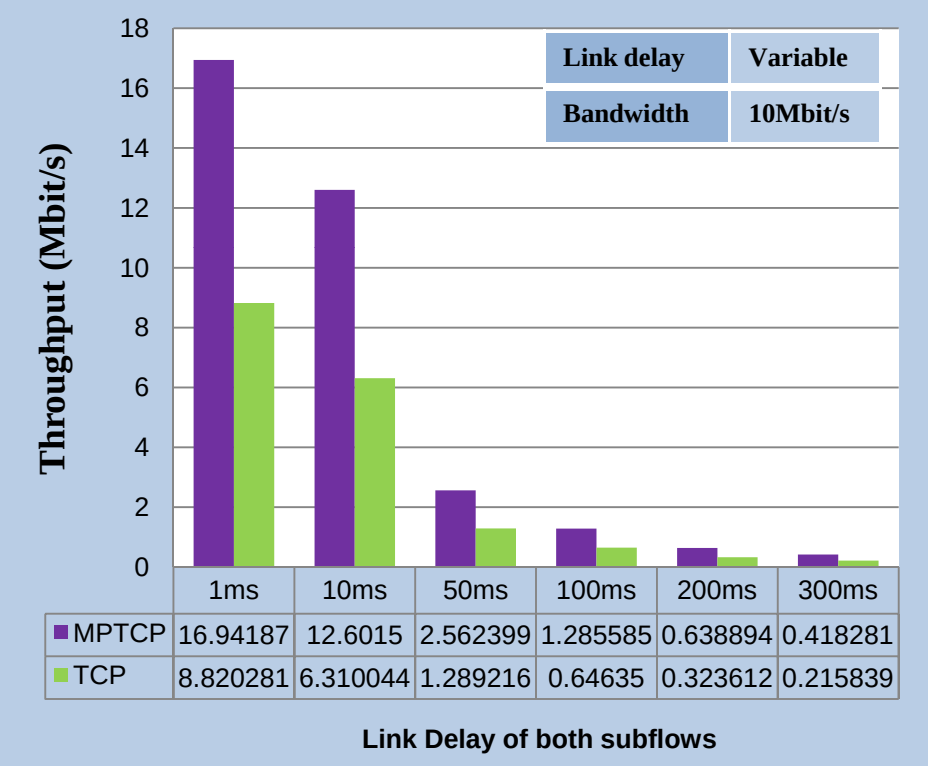


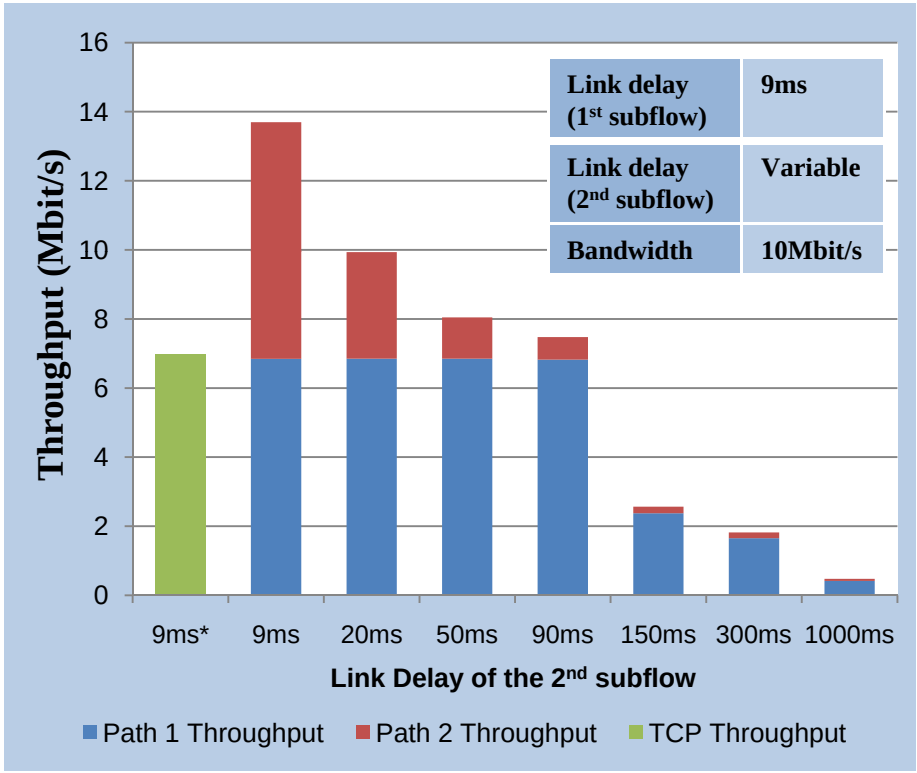
	50	50*	30	30*	10	10*	1	1*	0.1	0.1*
A2<-->B2	7.0101	7.0514	6.2231	6.3612	1.9773	4.2072	0.2088	0.4235	0.0208	0.0423
A2<-->B1	7.016		6.2329		2.0462		0.224		0.0267	
A1<-->B2	7.0028		6.2139		2.2385		0.2353		0.0229	
A1<-->B1	7.0233	7.0313	6.2498	6.3812	2.1983	4.2197	0.1948	0.4247	0.0197	0.0424

Throughput of MPTCP is limited by the bandwidth of the common link.

Scenario 1: Performance Evaluation (2)

Comparison of Throughput in MPTCP connections
(Link Delay)





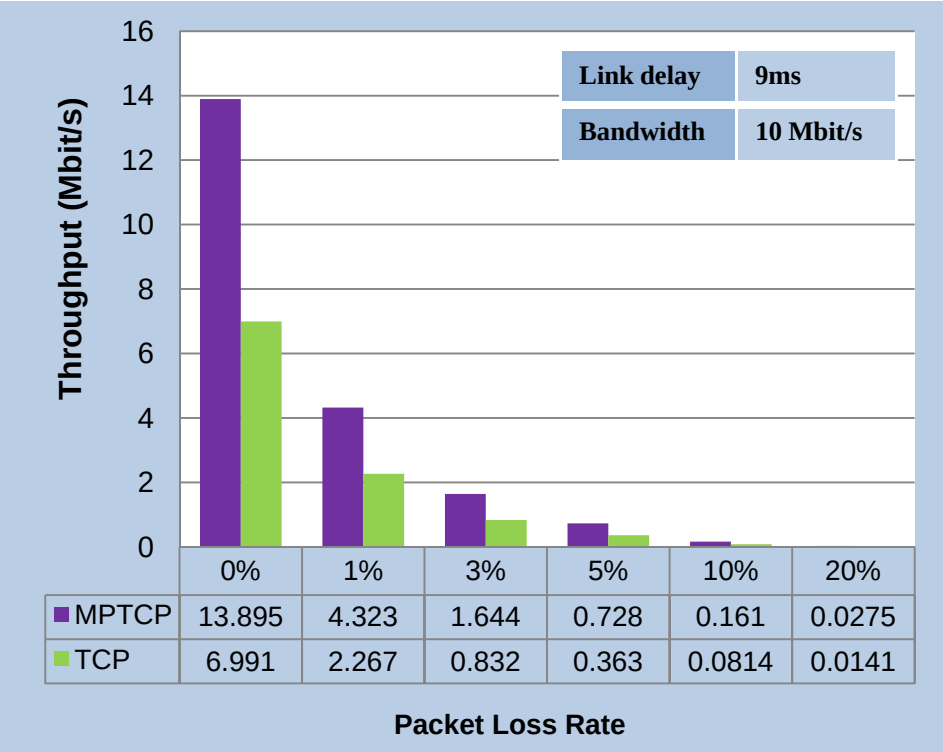
Two MPTCP subflows have **same link delay**

Two MPTCP subflows have **different link delay**

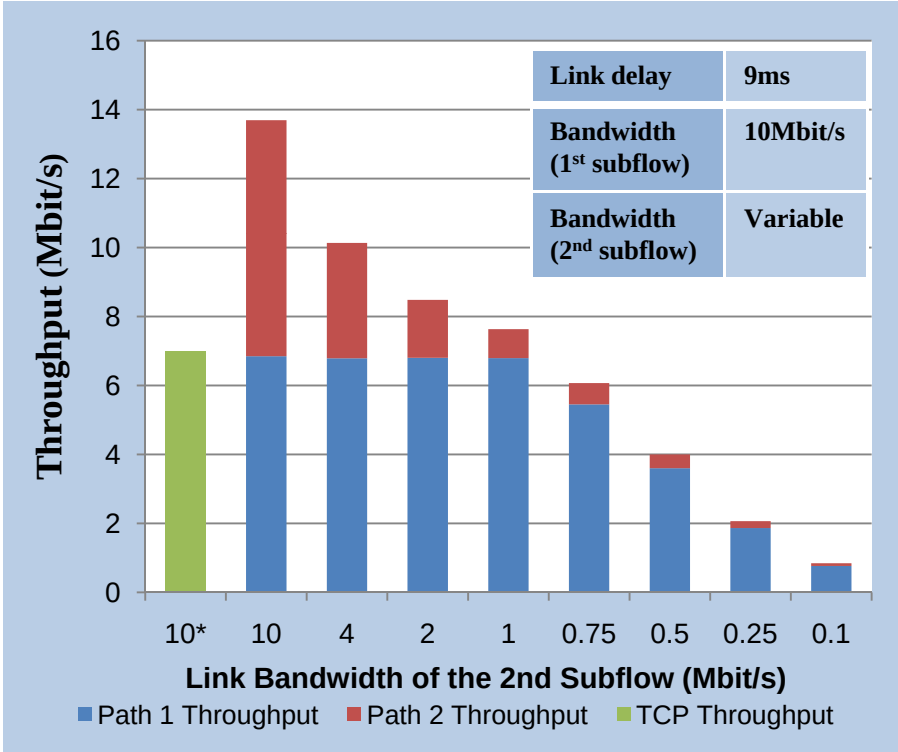
Scenario 1: Performance Evaluation (3)

Comparison of Throughput in MPTCP connections

Packet Loss Rate



Bandwidth



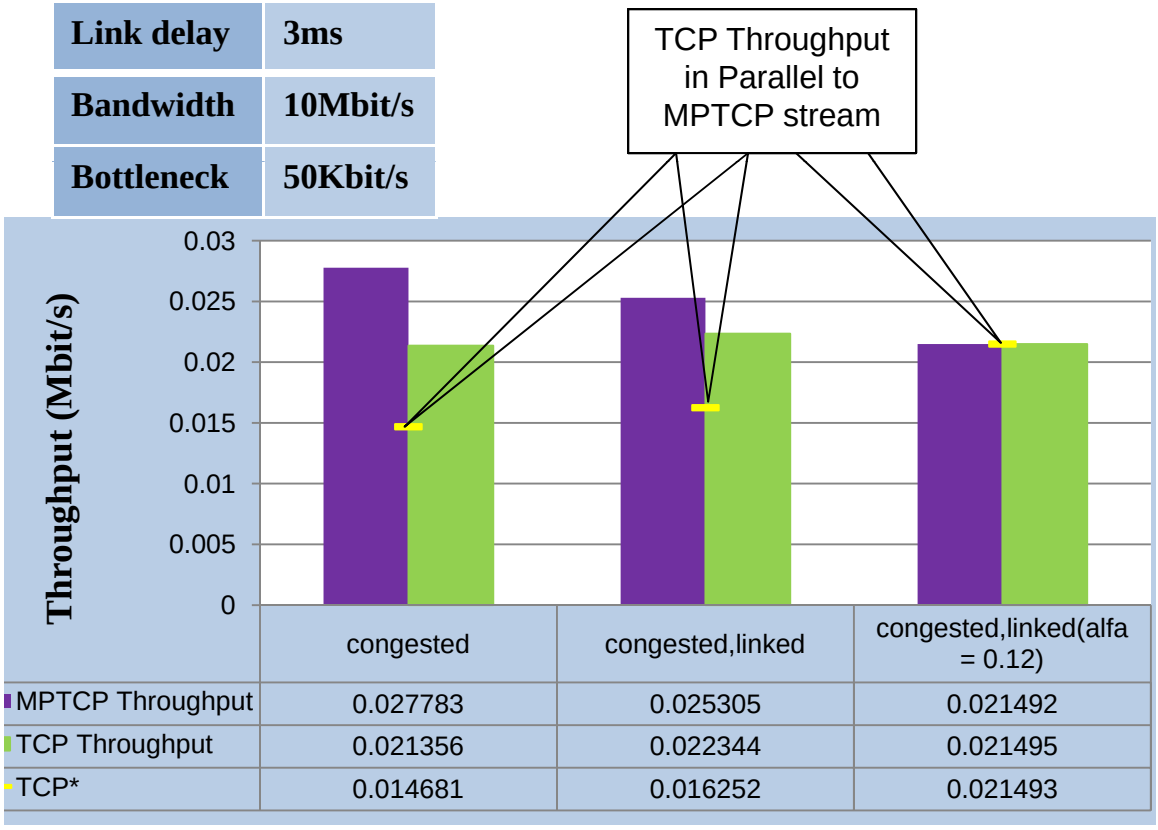
Two MPTCP subflows have **same packet loss rate**

Two MPTCP subflows have **different link bandwidth**

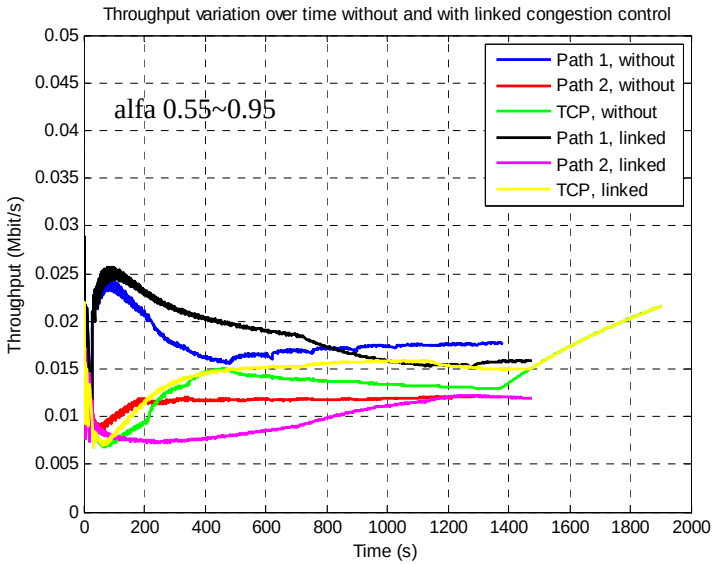
Scenario 2: Performance Evaluation

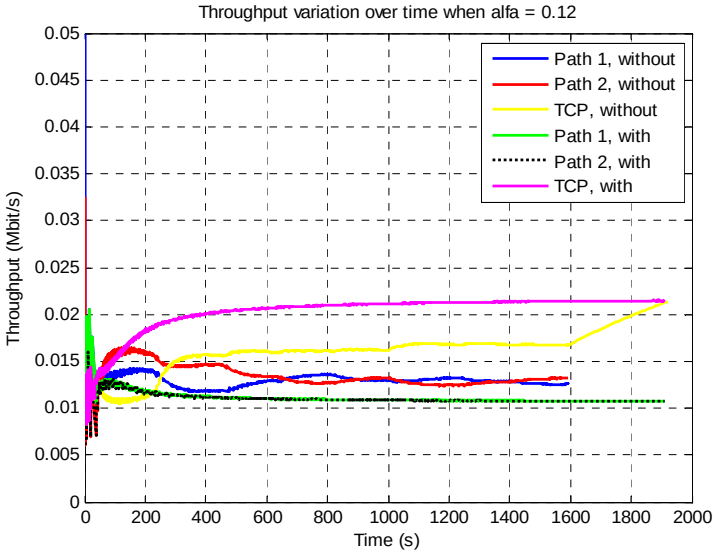
Throughput variation over time for two subflows of the MPTCP connection and the single-path TCP connection

Link delay	**3ms**
Bandwidth	**10Mbit/s**
Bottleneck	**50Kbit/s**



	congested	congested,linked	congested,linked(alfa = 0.12)
MPTCP Throughput	0.027783	0.025305	0.021492
TCP Throughput	0.021356	0.022344	0.021495
TCP*	0.014681	0.016252	0.021493





Conclusion and Outlook

- MPTCP solutions adhere to the requirement goals of the multipath TCP architecture
  - Increase throughput, more resilient and move congestion away from the congested paths
  - Offers reliable, in-order transport being transparent to applications
- Different proposed multipath TCP solutions within the MPTCP WG
  - span different design choices (option, payload & hybrid) for exchange of multipath signaling information
  - have different mechanisms for multipath initiation
  - PLMT comes around as the easiest solution for deployment and testing along with the possibility of extensions
- There is no connection level congestion control
  - Linked congestion control is proposed within the IETF for coupled (subflow) congestion control, though it is not completely fair to the legacy TCP connections while sharing a bottleneck link.
  - Linked congestion control cannot be implemented in the user-space
- How many subflows can be opened for a MPTCP flow and its dependance on different parameters needs to be further investigated
- Dependence on the send and receive connection level buffer was identified. The criteria when to close an under performing subflow needs to be further investigated.
- MPTCP-specific APIs for MPTCP-aware applications to control the setup of a multipath TCP connection and operation.

References

[1] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[2] Wischik, D., Handley, M., and M. Bagnulo Braun, "The Resource Pooling Principle", ACM SIGCOMM CCR vol. 38 num. 5, pp. 47-52, October 2008.

[3] Ford, A., Raiciu, C., Barre, S., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development“, draft-ietf-mptcp-architecture-01 (work in progress), June 2010.

[4] Carpenter, B. and S. Brim, "Middleboxes: Taxonomy and Issues", RFC 3234, February 2002.

[5] Ford, B. and J. Iyengar, "Breaking Up the Transport Logjam“, ACM HotNets, October 2008.

[6] C. Raiciu, M. Handley, and D. Wischik, “Practical Congestion Control for Multipath Transport Protocols.”

[7] Raiciu, C., Handley, M., and D. Wischik, "Coupled Multipath-Aware Congestion Control", draft-ietf-mptcp-congestion-00 (work in progress), July 2010.

[8] Ford, A., Raiciu, C., and M. Handley, "TCP Extensions for Multipath Operation with Multiple Addresses“, draft-ietf-mptcp-multiaddressed-00 (work in progress), June 2010.

[9] Singh, A., and M. Scharf, "PayLoad Multi-connection Transport using Multiple Addresses”, draft-singh-mptcp-plmt-00 (work in progress), August 2010.

[10] Scharf,M., "Multi-Connection TCP (MCTCP) Transport“, draft-scharf-mptcp-mctcp-01 (work in progress), July 2010.

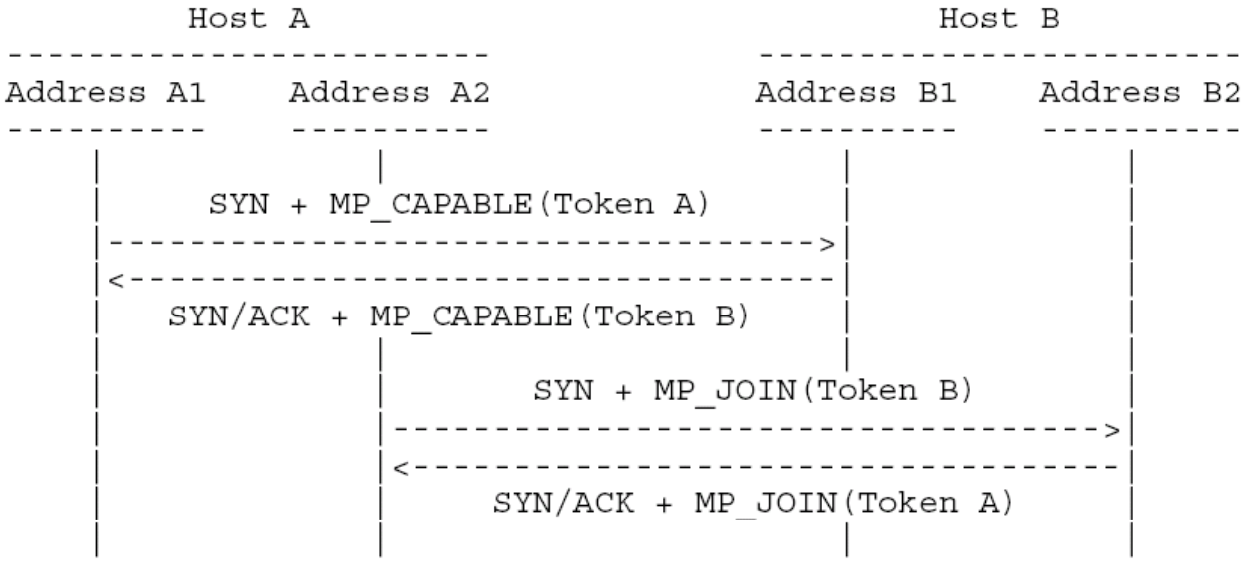
[11] Barre, S., MPTCP Linux Kernel Implementation Architecture, https://scm.info.ucl.ac.be/trac/mptcp/

ANY QUESTIONS ?

BACK UP

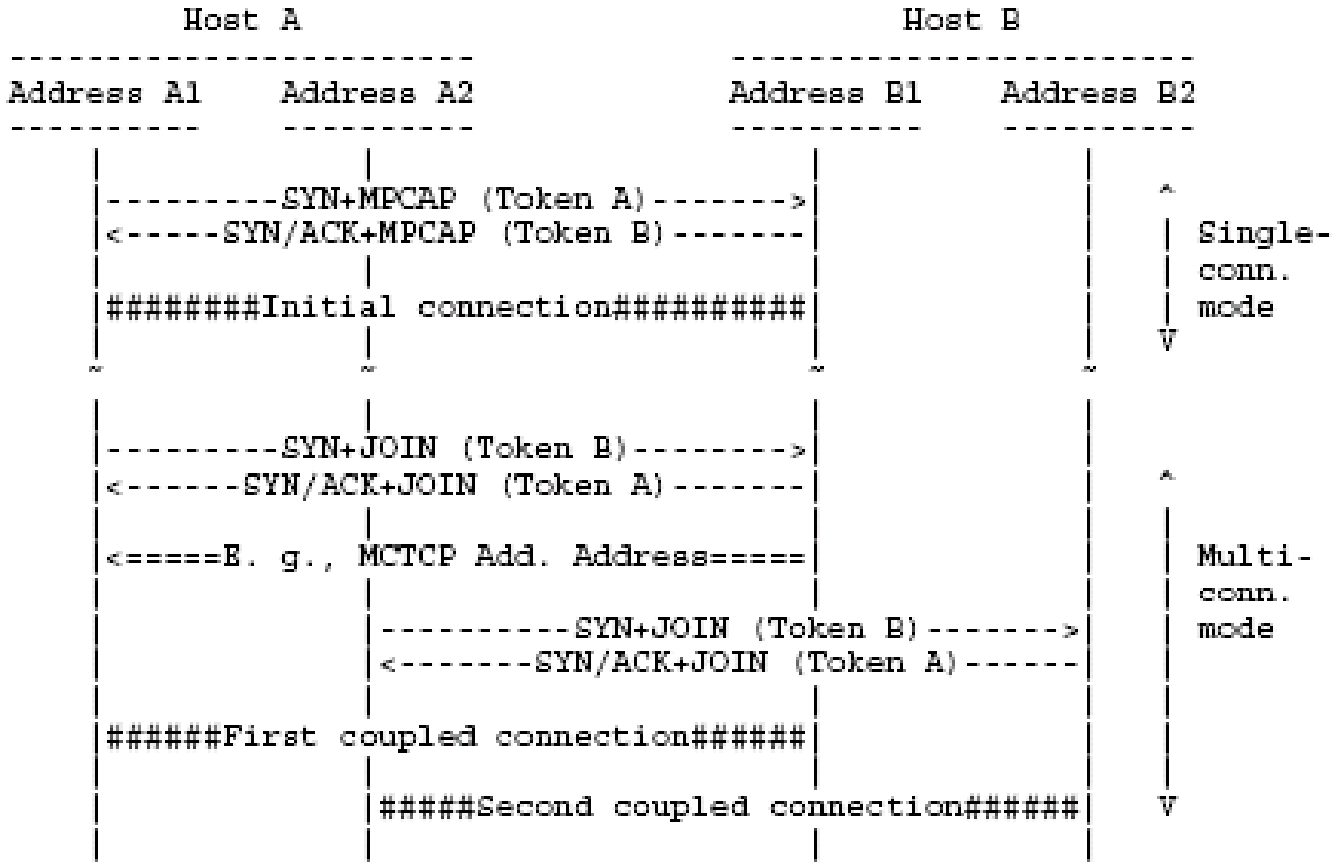
MPTCP Operation

- When a new TCP flow is started, the multipath capable option is exchanged between the endpoints within the SYN/ACK packet.
- Additional subflows are only opened after the initial handshake succeeds.
- The stack checks to see if it has multiple addresses that have routes to the destination (address A2). To get around NATs, addresses are also signaled explicitly to the remote end using TCP options (address B2).
- Subflows are created with the usual three way handshake with SYN packets carrying a “Join” option and a connection identifier.



MCTCP Operation

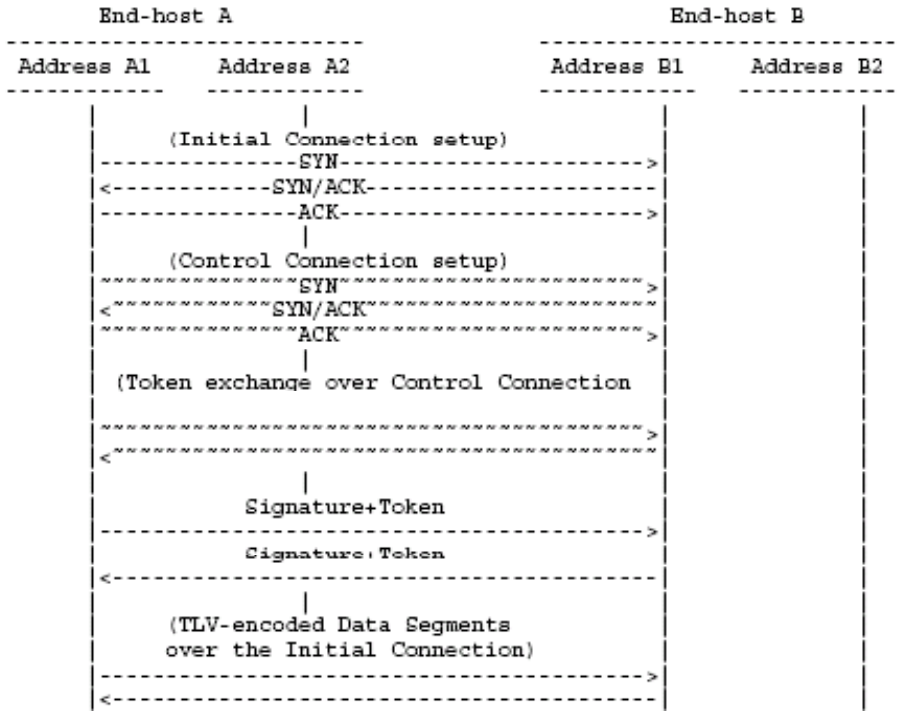
- MCTCP encodes control information, as far as possible, in the payload of the TCP connections and therefore requires only minor changes in the TCP implementations, and it is transparent in the single-path case.



PLMT Operation

- PLMT operates as an additional protocol layer between the network stack and the application layer and therefore is an example for a multipath mechanism that could possibly be realized entirely in the user-space of an operating system.

Parallel Control Connection & Initial Connection setup



Control Connection is setup Later w.r.t. Initial Connection