MultiPath TCP: From Theory to Practice

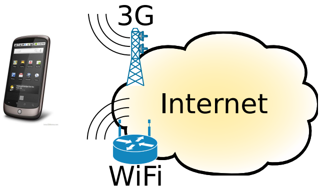
Sébastien Barré
Christoph Paasch
Olivier Bonaventure

9 mai 2011

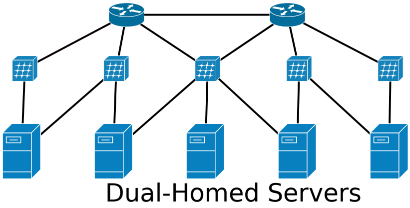
http ://inl.info.ucl.ac.be/mptcp/

MultiPath TCP

- Mobile devices can connect to the Internet via different interfaces

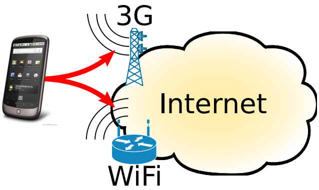


- Data-centers have a large redundant infrastructure

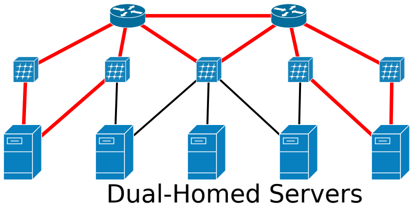


MultiPath TCP

- Mobile devices can connect to the Internet via different interfaces



- Data-centers have a large redundant infrastructure



MultiPath TCP

State of the Art

- TCP is used for 95% of the Internet communications
- A single TCP connection cannot be used across different interfaces.

MultiPath TCP (short MPTCP)

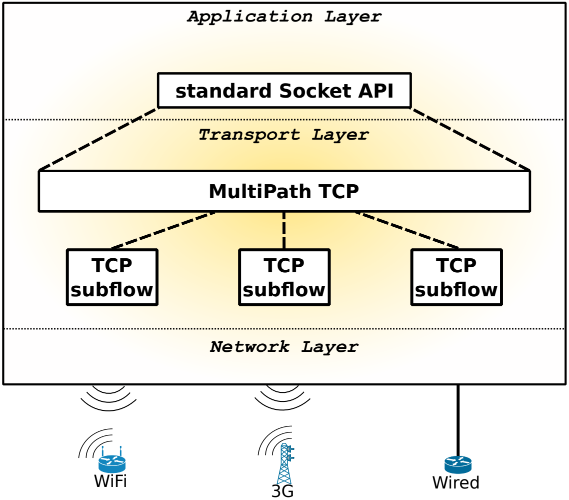
- MPTCP allows a single data-connection to use several interfaces simultaneously.
- Allows failover from one interface to another (e.g., mobile client).
- Increases the bandwidth due to resource pooling.

MultiPath TCP : From Theory To Practice

S. Barré, C. Paasch, O. Bonaventure

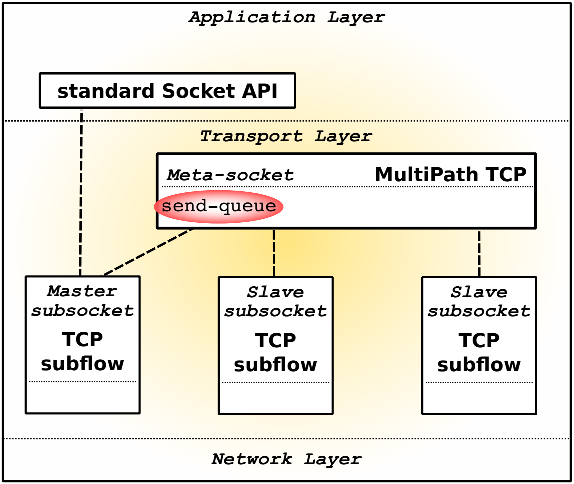
- We implemented MultiPath TCP in the Linux Kernel.
- Solved several design challenges that needed to be considered.
- Evaluated and measured the impact of our design choices in a testbed.
- Live-Demo of MPTCP at the end of this presentation.

MultiPath TCP - Architecture - From Theory



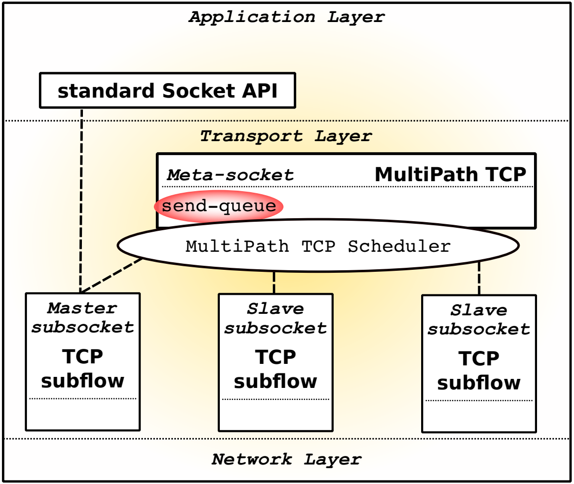
MultiPath TCP - Architecture - To Practice

Sending packets over MultiPath TCP



MultiPath TCP - Architecture - To Practice

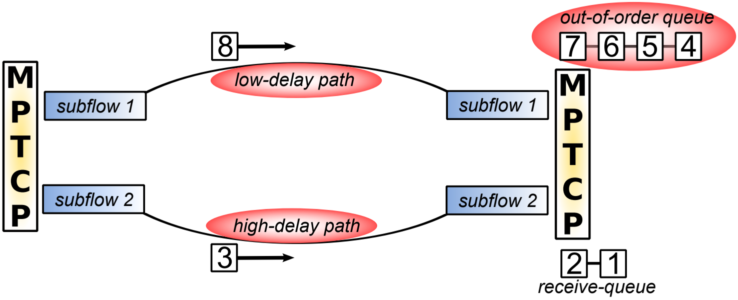
Sending packets over MultiPath TCP



MultiPath TCP - Architecture - To Practice

Receiving packets over MultiPath TCP

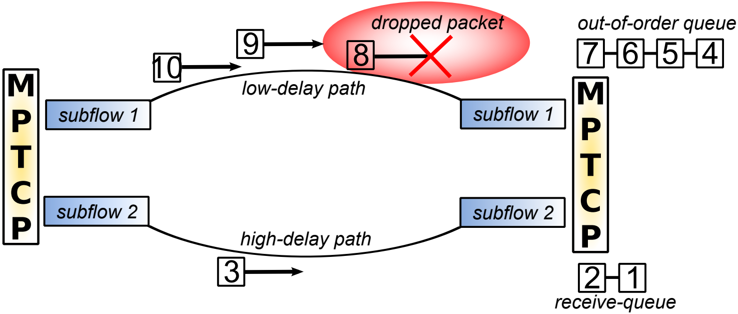
Packets can be reordered at the data-level due to delay-differences.



MultiPath TCP - Architecture - To Practice

Receiving packets over MultiPath TCP

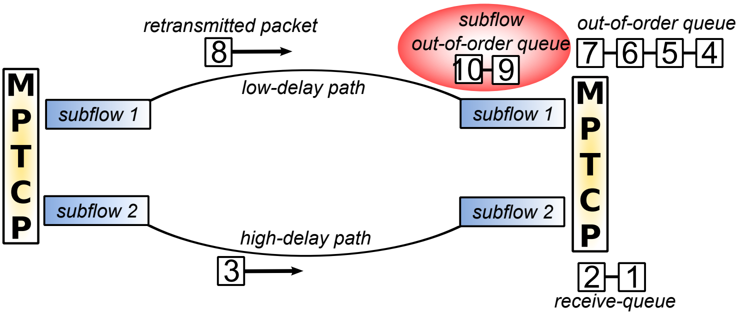
A loss at the subflow-level (or network-reordering) can also cause reordering at the subflow-level



MultiPath TCP - Architecture - To Practice

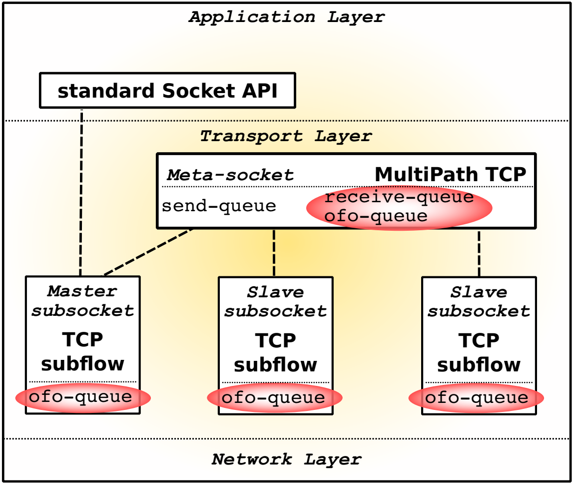
Receiving packets over MultiPath TCP

Subflow-level out-of-order queues are necessary to handle the retransmission at the subflow-level



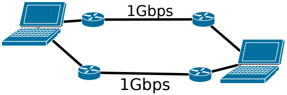
MultiPath TCP - Architecture - To Practice

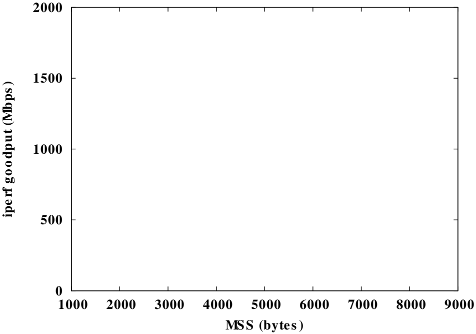
Receiving packets over MultiPath TCP



MultiPath TCP - Architecture - To Practice

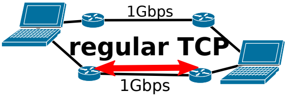
Interconnected testbed with two separate paths at 1Gbps

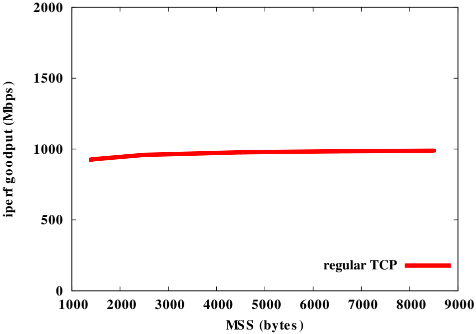




MultiPath TCP - Architecture - To Practice

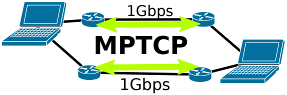
regular TCP can only use one single path

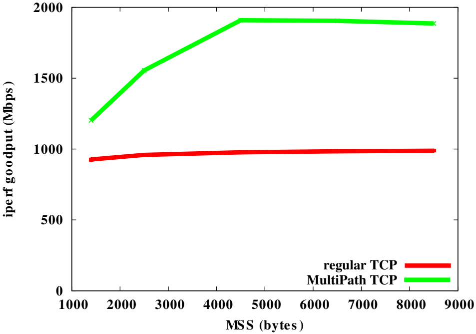




MultiPath TCP - Architecture - To Practice

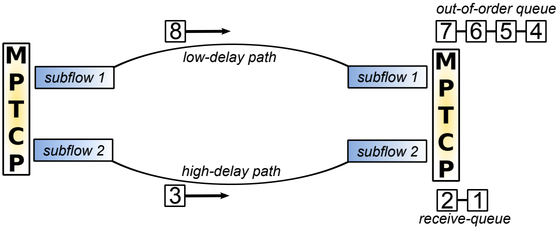
MultiPath TCP uses both paths simultaneously





MultiPath TCP - Receive-Buffer - From Theory

- **Regular TCP** : receive-buffer should be twice the BDP
  - Support a fast-retransmit
  - Support network-level reordering
- **MultiPath TCP** : Higher reordering possible due to delay-differences of the paths.



MultiPath TCP - Receive-Buffer - To Practice

⇒A subflow on a slow path (high RTT) may block a subflow on a fast path from transmitting.

$$RTT_{max}$$

MultiPath TCP - Receive-Buffer - To Practice

⇒A subflow on a slow path (high RTT) may block a subflow on a fast path from transmitting.

⇒We need to cope with one fast-retransmit and network-level reordering on this slow path

$$RTT_{max} * 2$$

MultiPath TCP - Receive-Buffer - To Practice

⇒A subflow on a slow path (high RTT) may block a subflow on a fast path from transmitting.

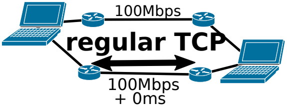
⇒We need to cope with one fast-retransmit and network-level reordering on this slow path

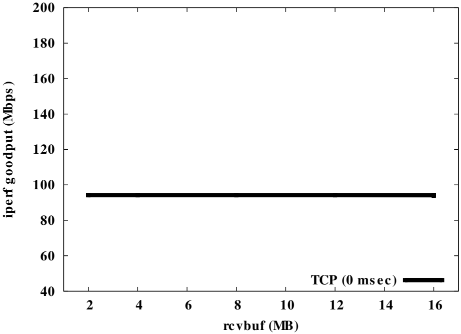
⇒During this time, all other subflows should be able to transmit at full speed (BW_i = bandwidth of subflow i)

$$RTT_{max} * 2 * \sum_{i \in subflows} BW_i$$

MultiPath TCP - Receive-Buffer - To Practice

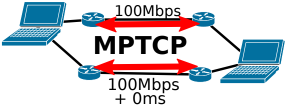
Regular TCP only gets 100Mbps of goodput

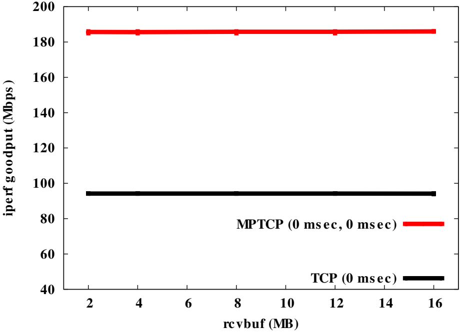




MultiPath TCP - Receive-Buffer - To Practice

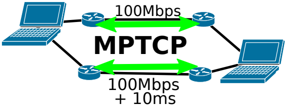
MultiPath TCP doubles the goodput

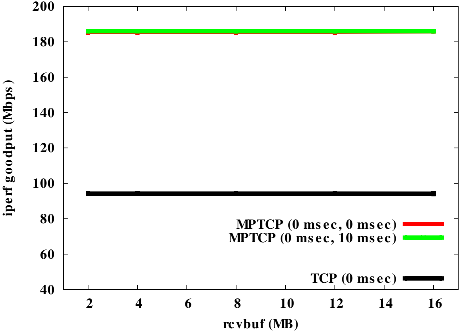




MultiPath TCP - Receive-Buffer - To Practice

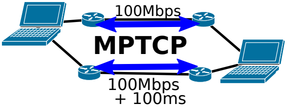
10ms and 100ms of difference has not a huge influence

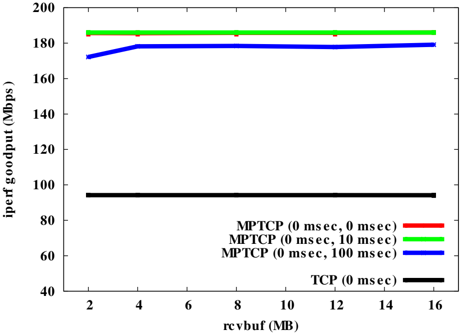




MultiPath TCP - Receive-Buffer - To Practice

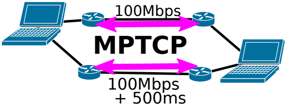
10ms and 100ms of difference has not a huge influence

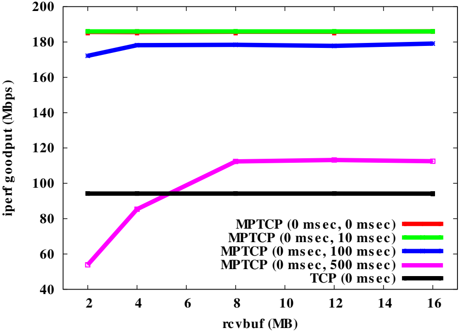




MultiPath TCP - Receive-Buffer - To Practice

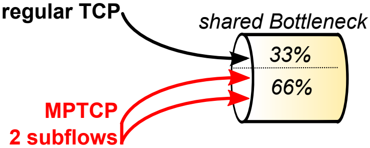
500ms of difference affects the goodput significantly

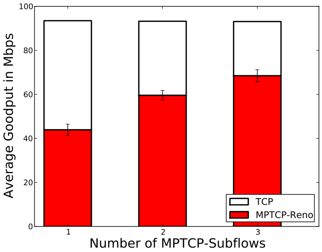




MultiPath TCP - Congestion Control

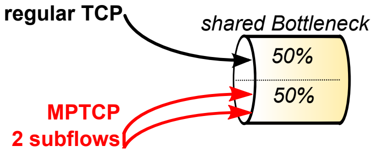
Several subflows will "eat up" regular TCP

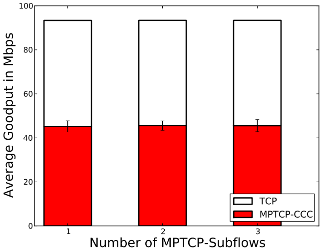




MultiPath TCP - Congestion Control

Our implementation of the Coupled Congestion Control [1] is fair to regular TCP across shared bottlenecks and avoids the paths with congested links.



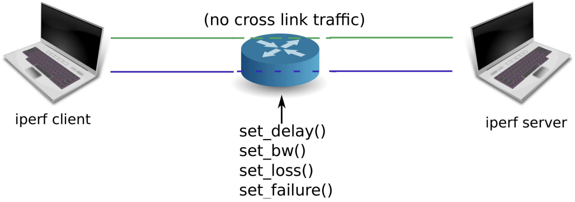


1. D. Wischik, C. Raiciu, A. Greenhalgh, M. Handley. *"Design, implementation and evaluation of congestion control for multipath TCP"*. NSDI'2011

MultiPath TCP - Live-Demo

- Access **http ://inl.info.ucl.ac.be/mptcp** to download our implementation (source-code, Live-CD,...).
- Contributions are always welcome ! ! !

Setup for the Live-Demo :



MultiPath TCP - Congestion Control - Theory

Coupled Congestion Control [2]

For each ack on subflow i :

$$cwnd_i = cwnd_i + min(\alpha / cwnd_{tot}, 1/cwnd_i)$$

$$\alpha = cwnd_{tot} \frac{\max_i(\frac{cwnd_i * mss_i^2}{RTT_i^2})}{(\sum_i \frac{cwnd_i * mss_i}{RTT_i})^2}$$

2. D. Wischik, C. Raiciu, A. Greenhalgh, M. Handley. *"Design, implementation and evaluation of congestion control for multipath TCP"*. NSDI'2011

MultiPath TCP - Congestion Control - Practice

Congestion Control in the Linux Kernel

Limitations :

- Congestion window is expressed in packets
- Linux Kernel cannot handle floating point numbers

Coupled Congestion Control implementation

- Count the number of acknowledged packets per subflow in $cwnd\_cnt_i$.
- Increase congestion window by one, if $cwnd\_cnt_i > max(tot_{cwnd}/\alpha, 1/cwnd_i)$
- Calculate alpha by minimizing the number of divisions and scaling the remaining ones :

$$\alpha = cwnd_{tot} \frac{cwnd_{max} * scale_{num}}{(\sum_i \frac{rtt_{max} * cwnd_i * scale_{den}}{rtt_i})^2}$$