Improving Multipath TCP

PhD Thesis - Christoph Paasch
The Internet is like a map ...
... highly connected
Communicating over the Internet
Multipath communication
Multipath communication
Multipath communication
Multipath communication

- Faster communication (resource pooling)
- Seamless handover (resilience to failures)
Multipath TCP
Multipath TCP

Web-Server

IP: A

IP: B
Multipath TCP

Web-Server

IP: A

IP: B

Dst: A Seq: 1

Dst: A Seq: 2

Dst: A Seq: 3

Dst: A Seq: 4

Dst: B Seq: 5

Dst: B Seq: 6

Dst: B Seq: 7

Dst: B Seq: 8
Multipath TCP

IP: A

IP: B

Web-Server

Internet

WiFi

3G/LTE

Dst: A
Seq: 1

Dst: A
Seq: 2

Dst: A
Seq: 3

Dst: A
Seq: 4

Dst: B
Seq: 5

Dst: B
Seq: 6

Dst: B
Seq: 7

Dst: B
Seq: 8
Multipath TCP
Multipath TCP handshake

Src: A, Dst: S
SYN
MPTCP: ID - X

Src: S, Dst: A
SYN + ACK
MPTCP: ID - Y
Multipath TCP handshake

Src: B, Dst: S
SYN
MP_JOIN: ID - Y

Src: S, Dst: B
SYN + ACK
MP_JOIN

Src: B, Dst: S
SYN
MP_JOIN: ID - Y
Transmit data with Multipath TCP
Multipath TCP

Can Multipath TCP be used on the Internet?

- Performance?
- Can it be implemented?
- Could it be designed differently?
Improving Multipath TCP
Improving Multipath TCP

- Implementing Multipath TCP
- Evaluating Transport Protocols
- Multipath TCP “in action”
Improving Multipath TCP

- Implementing Multipath TCP
- Evaluating Transport Protocols
- Multipath TCP “in action”
Implementing Multipath TCP
Goals of the implementation

1. Minimize performance impact on regular TCP

2. Reduce complexity within regular TCP

3. Achieve high performance for Multipath TCP
Why a Linux Kernel implementation?

- Tightly integrated into the TCP-stack
- Best performance
- Reactive to changes in path characteristics
Implementing Multipath TCP

… and how it fits inside the Linux Kernel
(based on a prototype from Sébastien Barré)
Architecture of Multipath TCP

User-Space

Application Layer

standard Socket API

Transport Layer

Multipath TCP

Network Layer

TCP subflow
send-queue

TCP subflow
send-queue

send-queue
receive-queue
Data structures of regular TCP

Application Layer

socket

tcp_sock

Network Layer

MPTCP-layer

subflow-layer
Data structures of Multipath TCP

Application Layer

socket

tcp_sock
“meta-socket”

mptcp_tcp_sock

Network Layer

MPTCP-layer

mptcp_cb

subflow-layer

tcp_sock
TCP subflow

mptcp_tcp_sock
Linux Kernel Multipath TCP

... and its performance compared to regular TCP
MPTCP performance with Apache

100 simultaneous HTTP-Requests, total of 100,000

Graph showing MPTCP performance compared to regular TCP.
MPTCP performance with Apache

100 simultaneous HTTP-Requests, total of 100000

![Diagram showing MPTCP performance with Apache](image)
MPTCP performance with Apache

100 simultaneous HTTP-Requests, total of 100000

1 Gbps

Requests per second

Transfer-size [KB]

regular TCP
TCP with link-bonding
MPTCP
Improving Multipath TCP

- Implementing Multipath TCP
- Evaluating Transport Protocols
- Multipath TCP “in action”
Evaluating Transport Protocols
Evaluating Multipath TCP

*Diagram showing a 3D graph with axes for RTT, Capacity, and Bufferbloat.*
Evaluating Multipath TCP
Evaluating Multipath TCP

“How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP”. C. Raiciu, et al. NSDI’12
Evaluating Multipath TCP

Evaluating Multipath TCP

Evaluating Multipath TCP
Evaluating Multipath TCP
Experimental Design

... a scientific approach to evaluation
The planned approach to evaluation

1. Define the objective
2. Decide the factors
3. Design the experiment
Design the experiment

Space-Filling Designs
Evaluating Multipath TCP’s resource pooling

… using the Experimental Design approach
1. Objective

Quantify Multipath TCP’s resource pooling capabilities

Aggregation Benefit

0 Mb/s  Same as best path  Perfect aggregation
-1  0  1

2. Domains of the factors

<table>
<thead>
<tr>
<th></th>
<th>Capacity</th>
<th>RTT</th>
<th>Buffering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-BDP</td>
<td>0.1 to 100 Mbps</td>
<td>0 to 50 ms</td>
<td>0 to 100 ms</td>
</tr>
<tr>
<td>High-BDP</td>
<td>0.1 to 100 Mbps</td>
<td>0 to 400 ms</td>
<td>0 to 2000 ms</td>
</tr>
</tbody>
</table>
3. Experiment design

6-Dimensional Space

~200 parameter-sets in a space-filling design

~4 hours of experiments
Visualizing the output
Congestion Control

Detecting the link’s capacity

- Loss-based (LIA)
- Delay-based (wVegas)
Resource Pooling (LIA)

Resource Pooling (wVegas)

Improving Multipath TCP

- Implementing Multipath TCP
- Evaluating Transport Protocols
- Multipath TCP “in action”
Multipath TCP

- Supported across a middleboxes
- Handover traffic from WiFi to 3G
- Generic Scheduling Infrastructure
Supported across middleboxes
Multipath TCP

- Supported across a middleboxes
- Handover traffic from WiFi to 3G
- Generic Scheduling Infrastructure
Scheduling

Web-Server

Low-Delay

High-Delay

Dst: A
Seq: 1

Dst: A
Seq: 2

Dst: A
Seq: 3

Dst: A
Seq: 4

Dst: A
Seq: 5
Handover
High Performance

- Zero-copy support
- Flow-to-core affinity
- Hardware offloading supported
The fastest TCP connection

10 Gig Links
The fastest TCP connection
Conclusion
Conclusion

- It is **scalable** in the Linux Kernel
- *Experimental Design* allows to better evaluate transport layer protocols
- Works "*in action*" across the Internet
Conclusion

● Multipath TCP works mostly well in heterogeneous environments

● Possible design evolution for more flexibility
Thank you!