Transitioning to MPTCP with efficient Protocol Converters

Gregory Detal <gregory.detal@uclouvain.be>
Christoph Paasch <christoph.paasch@uclouvain.be>

IP Networking Lab - UCLouvain
Need for Protocol Converters

- Transition to MPTCP will take several years
- Encourage early adopters by deploying protocol converters
Need for Protocol Converters
Our Design

- Translate MPTCP to TCP (or vice-versa)
- Supports multiple subflows

- Full In-Kernel implementation (3K LOC)
- High performance
  - Minimized allocations
  - Minimized data-copying
Our Design

Protocol converter

- MPTCP socket
  - sk_receive_queue
  - sk_write_queue
  - out_of_order_queue
- TCP socket
  - sk_receive_queue
  - sk_write_queue
  - out_of_order_queue

Client

- MPTCP side

Server

- TCP side

Forward

Move
Performance evaluation
Protocol Converter vs. other solutions

- **Layer 7**: Squid or Custom Application
- **Layer 4**: Protocol Converter
- **Layer 3**: Router (no conversion)
Apachebench benchmark

- 100 parallel clients
- HTTP-get for files of varying size
- Up to 100000 Requests
- Measuring the number of requests per second

MPTCP-client  Protocol Converter  TCP-server
Apache benchmark
Apachebenchmark

Requests per second vs. Request size [KB]
Apache benchmark

![Graph showing requests per second vs request size in KB, comparing Router and Squid.]
Apachebenchmark
Forwarding at 10Gbps

MPTCP-client -> Protocol Converter -> TCP-server
Forwarding at 10Gbps
Forwarding at 10Gbps
Forwarding at 10Gbps

![Graph showing goodput vs. maximum segments size (MSS)]
Forwarding at 10Gbps

![Graph showing goodput vs. maximum segments size (MSS) in bytes]

- **Router**
- **Protocol Converter**
- **Read/write App.**
Video
Remote-Desktop to a TCP-only Windows server, passing by the Protocol Converter.

Connection is not lost after unplugging the Ethernet cable.