Options vs. Payload Encoding: MCTCP’s Perspective
draft-scharf-mptcp-mctcp-01

Michael Scharf <michael.scharf@alcatel-lucent.com>

IETF 78, July 2010
General Thoughts on Options and Payload Encoding

Overview

- **Alternatives** with similar service like draft-ietf-mptcp-multiaddressed-01
  - **MCTCP** is a hybrid solution that uses both option and payload encoding
  - Combines features of a TCP extension and an app protocol (“best of both”)

![Diagram]

- This presentation gives an overview only
  - Not all details addressed, e.g., feasibility of a user space solution [1] [2]
  - **Terminology** according to draft-ietf-mptcp-architecture-01 to simplify the discussion

General Thoughts on Options and Payload Encoding
Requirements and Constraints

- From charter: **Usable without significant changes** to Internet infrastructure
  - **Only one subflow**: Bytestream should probably be identical to TCP
  - **More than one subflow**: Does the bytestream format matter?

- **Possible solution**: Use payload encoding if there is more than subflow
  - **Rationale**: Middleboxes then cannot parse the app data in all encoding variants
  - **Type-length-value (TLV) framing** reasonable, alternatives possible (e.g., MIME-like)
  - Similar to protocols such as TLS

- TCP options can hardly be avoided due to **backward compatibility**
  - In **SYNs** to identify initial and follow-up subflows ➔ space issue
  - **Outside SYNs** on initial subflow ➔ only required to detect multihomed servers

Question: What multipath protocol design **minimizes the use of TCP options**?
MCTCP’s Hybrid Solution
Message Sequence Chart

Host A
daddr. a1
addr. a2

Host B
daddr. b1
addr. b2

Single connection mode

Data transfer over initial subflow: byte stream, identically to standard TCP

Change over

Multi connection mode

Data transfer over coupled subflow #1: data in TLV encoded messages

Initial subflow idle as a fallback

Address announce msg.

Data transfer over coupled subflow #2: data in TLV encoded messages

→ Payload encoding used on all follow-up subflows (“coupled connections”)

→ Initial subflow is kept established as fallback, e.g., if TLV is blocked

Initial subflow
→ Unmodified bytestream

Follow-up subflow #1 (“1st coupled connection”)
→ TLV encoding

Follow-up subflow #2 (“2nd coupled connection”)
→ TLV encoding
MCTCP’s Hybrid Solution
Advantage of Payload Encoding: Robustness

- **Unaffected by middleboxes** stripping options or dropping packets w. options
  - In the worst case, MCTCP is not enabled, and SYN must be retransmitted
  - A sender cannot safely determine middleboxes stripping options outside SYNs, in particular if routing changes

- **Change of routing**: Example (others exist, too)

  ![Diagram](image)

<table>
<thead>
<tr>
<th>Connection type</th>
<th>MPTCP</th>
<th>MCTCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Existing connections</td>
<td>Multipath transport</td>
<td>Multipath transport</td>
</tr>
<tr>
<td>before handover</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Existing connections</td>
<td>Break (fallback to single path?)</td>
<td>Multipath transport continues</td>
</tr>
<tr>
<td>after handover</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New connections after handover</td>
<td>Single path transport</td>
<td>Single path transport</td>
</tr>
</tbody>
</table>

- Are such middleboxes indeed a **relevant** issue?
  - Measurements report that such middleboxes are currently rare (<1%)
  - Fundamental question: Will middleboxes try to prevent MPTCP usage in future?
MCTCP’s Hybrid Solution
Further Advantages

- **Reliability**
  - Sender does not need a TCP option retransmission mechanism
  - Receiver does not need to deal with missing options, e.g., data without mapping

- **Extensibility**
  - Future protocol enhancements do not allocate TCP option code-points
  - No limitation to 40 byte
  - Less consumption of SYN option space (compared to draft-ietf-mptcp-multiaddressed)

- **No changes in TCP’s fast path processing** required
  - Existing offloading should work well
  - No segmentation issues due to variable MSS

- **Security**
  - Currently, same token mechanism like draft-ietf-mptcp-multiaddressed
  - A stronger authentication of follow-up subflows possible, e.g., with longer tokens
MCTCP’s Hybrid Solution

Drawbacks

- Payload on follow-up subflows includes **TLV headers**
  - Binary symbols on port 80 may **confuse DPI/IDS boxes that parse single packets** only
    - Note: Any middlebox reassembling the bytestream will be confused anyway
  - **MCTCP can fall back** to single-path TCP if TLV encoding is immediately blocked
  - **Other remedies:** Use another port, or, e.g., MIME-like encoding instead of TLV

- **Middleboxes must reassemble byte stream to access control information**
  - Parsing of addresses, data sequence numbers and/or data ACKs difficult
  - **No reasonable use case** for this identified so far
  - Such middleboxes might just want a simple way to **disable multipath transport**

- **One additional TCP connection** (e.g., 3 connections for 2 paths)
  - Initial subflow is kept established to **expose valid addresses** and as a fallback
  - **Alternative protocol design** could switch to TLV encoding on initial subflow

- **Minor semantic differences of options vs. payload** (e.g., URGENT flag)
MCTCP’s Hybrid Solution

Thoughts on Acknowledgements and Flow Control

- **Proper connection-level flow control** avoids deadlocks
- Data ACKs increase **robustness if memory is a constraint**
- Reliable, congestion-controlled transport **not always optimal for data ACKs**
- Data ACKs have **few benefit in some use cases** (e.g., data center use case)
  - If path failure is unlikely
  - If sender and receiver are not memory constrained
  - If there are no proactively acking middleboxes
- Suggestion: Data ACKs (+ connection-level flow control) as **optional feature**
  - Enabled by default
  - Can be turned off to optimize performance and to reduce processing overhead
  - Anyway, there can still be **negative/selective data acknowledgements**
Options vs. Payload

Summary

- Payload encoding is **more robust, extensible, and modular**
  - In-band acking results in **tradeoff of robustness vs. performance**
  - In some scenarios, **payload works, whereas options fail**
- Multipath transport is somehow **a shim layer on top of TCP connections**
  - Requires own addresses, own sequence numbers, own flow control, (maybe) own ACKs
  - These are actually characteristics of an **own protocol layer with own framing**
- Still, options vs. payload is not necessarily an **either-or question**
  - **MCTCP is a hybrid solution** combining payload encoding with options
  - Options are only used if they are really needed
- **MCTCP’s encoding fulfills the requirements** of the MPTCP architecture (draft-ietf-mptcp-architecture-01)