Transport Area Working Group
Internet-Draft
Intended status: Standards Track
Expires: 31 October 2025

M. Amend, Ed.
DT
A. Brunstrom
A. Kassler
Karlstad University
V. Rakocevic
City, University of London
S. Johnson
BT

29 April 2025

# DCCP Extensions for Multipath Operation with Multiple Addresses
draft-ietf-tsvwg-multipath-dccp-24

## Abstract

Datagram Congestion Control Protocol (DCCP) communications, as defined in RFC 4340, are inherently restricted to a single path per connection, despite the availability of multiple network paths between peers. The ability to utilize multiple paths simultaneously for a DCCP session can enhance network resource utilization, improve throughput, and increase resilience to network failures, ultimately enhancing the user experience.

Use cases for Multipath DCCP (MP-DCCP) include mobile devices (e.g., handsets, vehicles) and residential home gateways that maintain simultaneous connections to distinct network types, such as cellular and Wireless Local Area Networks (WLANs) or cellular and fixed access networks. Compared to existing multipath transport protocols, such as Multipath TCP (MPTCP), MP-DCCP is particularly suited for latency-sensitive applications with varying requirements for reliability and in-order delivery.

This document specifies a set of protocol extensions to DCCP that enable multipath operations. These extensions maintain the same service model as DCCP while introducing mechanisms to establish and utilize multiple concurrent DCCP flows across different network paths.

## Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

## Copyright Notice



## Table of Contents

## 1. Introduction

Datagram Congestion Control Protocol (DCCP) [RFC4340] is a transport protocol that provides bidirectional unicast connections of congestion-controlled unreliable datagrams. DCCP communications are restricted to one single path. Other fundamentals of the DCCP protocol are summarized in section 1 of [RFC4340], such as the reliable handshake process in section 4.7 and the reliable negotiation of features in section 4.5. These are an important basis for this document. This also applies to the DCCP sequencing scheme, which is packet-based (section 4.2), and the principles for loss and retransmission of features as described in more detail in section 6.6.3. This document specifies a set of protocol changes that add multipath support to DCCP; specifically, support for signaling and setting up multiple paths (a.k.a, "subflows"), managing these subflows, reordering of data, and termination of sessions.

Multipath DCCP (MP-DCCP) enables a DCCP connection to simultaneously establish a flow across multiple paths. This can be beneficial to applications that transfer large amounts of data, by utilizing the capacity/connectivity offered by multiple paths. In addition, the multipath extensions enable to tradeoff timeliness and reliability, which is important for low-latency applications that do not require guaranteed delivery services, such as Audio/Video streaming.

In addition to the integration into DCCP services, implementers or future specification could choose MP-DCCP for other use cases like 3GPP 5G multi-access solutions (e.g., Access Traffic Steering, Switching, and Splitting (ATSSS) specified in [TS23.501]) or hybrid access networks that either combine a 3GPP and a non-3GPP access or a fixed and cellular access between user-equipment/residential gateway and operator network. MP-DCCP can be used in these scenarios for load balancing, seamless session handover and bandwidth aggregation when non-DCCP traffic like IP, UDP or TCP is encapsulated into MP-DCCP. More details on potential use cases for MP-DCCP are provided in [multipath-dccp.org], [IETF105.Slides], and [MP-DCCP.Paper]. All these use cases profit from an Open Source Linux reference implementation provided under [multipath-dccp.org].

The encapsulation of non-DCCP traffic (e.g., UDP or IP) in MP-DCCP to enable the above-mentioned use cases is not considered in this specification. Also out of scope is the encapsulation of DCCP traffic in UDP to pass middleboxes (e.g., NATs, firewalls, proxies, intrusion detection systems (IDSs), etc) that do not support DCCP. A possible method is defined in [RFC6773] or is considered in [I-D.amend-tsvwg-dccp-udp-header-conversion] to achieve the same with less overhead.

MP-DCCP is based exclusively on the lean concept of DCCP. For traffic that is already encrypted or does not need encryption, MP-DCCP is an efficient choice as it does not apply its own encryption mechanisms. Also, the procedures defined by MP-DCCP, which allow subsequent reordering of traffic and efficient traffic scheduling, improve performance, as shown in [MP-DCCP.Paper], and take into account the interaction of the protocol with the further elements required for multi-path transport.

## 1.1. Multipath DCCP in the Networking Stack

MP-DCCP provides a set of features to DCCP; Figure 1 illustrates this layering. MP-DCCP is designed to be used by applications in the same way as DCCP with no changes to the application itself.

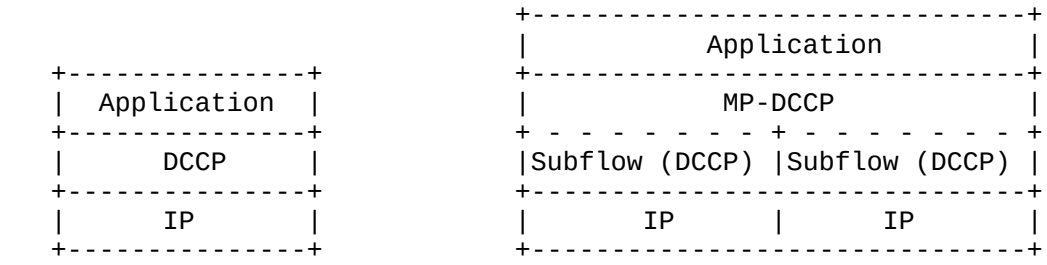



Figure 1: Comparison of standard DCCP and MP-DCCP protocol stacks

A CLI at the endpoint (or another method) could be used to configure and manage the DCCP Connections. This could be extended to also support MP-DCCP, but this specification does not define this.

## 1.2. Terminology

This document uses terms that are either specific for multipath transport as defined in [RFC8684] or are defined in the context of MP-DCCP, as follows:

Path: A sequence of links between a sender and a receiver, defined in this context by a 4-tuple of source and destination address and the source and destination ports. This definition follows [RFC8684] and is illustrated in the following two examples for IPv6 and IPv4, which each show a pair of sender IP-address:port and a pair of receiver IP-address:port, which together form the 4-tuple:

* IPv6: [2001:db8:3333:4444:5555:6666:7777:8888]:1234, [2001:db8:3333:4444:cccc:dddd:eeee:ffff]:4321

* IPv4: 203.0.113.1:1234, 203.0.113.2:4321

Subflow: A subflow refers to a DCCP flow transmitted using a specific path (4-tuple of source and destination address/port pairs) that forms one of the multipath flows used by a single connection.

(MP-DCCP) Connection: A set of one or more subflows, over which an application can communicate between two hosts. The MP-DCCP connection is exposed as single DCCP socket to the application.

Connection Identifier (CI): A unique identifier that is assigned to a multipath connection by the host to distinguish several multipath connections locally. The CIs must therefore be locally unique per host and do not have to be the same across the peers.

Host: An end host operating an MP-DCCP implementation, and either initiating or accepting an MP-DCCP connection.

'+': The plus symbol means concatenation of values.

In addition to these terms, within the framework of MP-DCCP, the interpretation of, and effect on, regular single-path DCCP semantics is discussed in Section 3.

### 1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

## 2. Operation Overview

DCCP transmits congestion-controlled unreliable datagrams over a single path. Various congestion control mechanisms have been specified to optimize DCCP performance for specific traffic types in terms of profiles denoted by a Congestion Control IDentifier (CCID). However, DCCP does not provide built-in support for managing multiple subflows within one DCCP connection. The extension of DCCP for Multipath-DCCP (MP-DCCP) is described in detail in Section 3.

At a high level of the MP-DCCP operation, the data stream from a DCCP application is split by MP-DCCP operation into one or more subflows which can be transmitted via different paths, for example using paths via different links. The corresponding control information allows the receiver to optionally re-assemble and deliver the received data in the originally transmitted order to the recipient application. This may be necessary because DCCP does not guarantee in-order delivery. The details of the transmission scheduling mechanism and optional reordering mechanism are up to the sender and receiver, respectively, and are outside the scope of this document.

A Multipath DCCP connection provides a bidirectional connection of datagrams between two hosts exchanging data using DCCP. It does not require any change to the applications. Multipath DCCP enables the hosts to use multiple paths with different 4-tuples to transport the packets of an MP-DCCP connection. MP-DCCP manages the request, set-up, authentication, prioritization, modification, and removal of the DCCP subflows on different paths as well as the exchange of performance parameters.
The number of DCCP subflows can vary during the lifetime of a Multipath DCCP connection. The details of the path management decisions for when to add or remove subflows are outside the scope of this document.

The Multipath Capability for MP-DCCP is negotiated with a new DCCP feature, as specified in Section 3.1. Once negotiated, all subsequent MP-DCCP operations for that connection are signalled with a variable length multipath-related option, as described in Section 3. All MP-DCCP operations are signaled by Multipath options described in Section 3.2. Options that require confirmation from the remote peer are retransmitted by the sender until confirmed or until confirmation is no longer considered relevant.

The following sections define MP-DCCP behavior in detail.

## 2.1. MP-DCCP Concept

Figure 2 provides a general overview of the MP-DCCP working mode, whose main characteristics are summarized in this section.

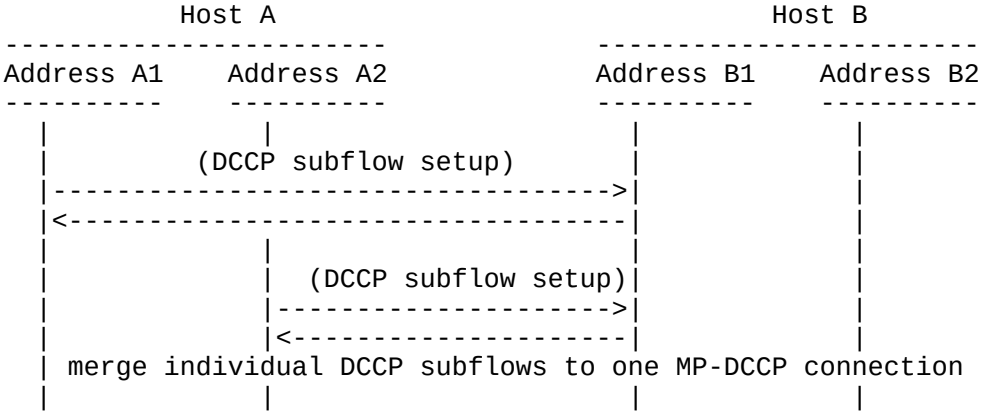



Figure 2: Example MP-DCCP usage scenario

* An MP-DCCP connection begins with a 4-way handshake, between two hosts. In Figure 2, an MP-DCCP connection is established between addresses A1 and B1 on Hosts A and B. In the handshake, a Multipath Capable feature is used to negotiate multipath support for the connection. Host specific keys are also exchanged between Host A and Host B during the handshake. The details of the MP-DCCP handshaking procedure is described in Section 3.3. MP-DCCP does not require both peers to have more than one address.

* When additional paths and corresponding addresses/ports are available, additional DCCP subflows can be created on these paths and attached to the existing MP-DCCP connection. An MP_JOIN option is used to connect a new DCCP subflow to an existing MP-DCCP connection. It contains a Connection Identifier during the setup of the initial subflow and is exchanged in the 4-way handshake for the subflow together with the Multipath Capable feature. The example in Figure 2 illustrates creation of an additional DCCP subflow between Address A2 on Host A and Address B1 on Host B. The two subflows continue to provide a single connection to the applications at both endpoints.

* MP-DCCP identifies multiple paths by the presence of multiple addresses/ports at hosts. Combinations of these multiple addresses/ports indicate the additional paths. In the example, other potential paths that could be set up are A1<->B2 and A2<->B2. Although the additional subflow in the example is shown as being initiated from A2, an additional subflow could alternatively have been initiated from B1 or B2.

* The discovery and setup of additional subflows is achieved through a path management method including the logic and details of the procedures for adding/removing subflows. This document describes the procedures that enable a host to initiate new subflows or to signal available IP addresses between peers. However, the definition of a path management method, in which sequence and when subflows are created, is outside the scope of this document. This method is subject to a corresponding policy and the specifics of the implementation. If an MP-DCCP peer host wishes to limit the maximum number of paths that can be maintained (e.g. similar to that discussed in section 3.4 of [RFC8041]), the creation of new subflows from that peer host is omitted when the threshold of maximum paths is exceeded and incoming subflow requests MUST be rejected.

* Through the use of multipath options, MP-DCCP adds connection-level sequence numbers and exchange of Round-Trip Time (RTT) information to enable optional reordering features. As a hint for scheduling decisions, a multipath option that allows a peer to indicate its priorities for what path to use is also defined.

* Subflows are terminated in the same way as regular DCCP connections, as described in ([RFC4340], Section 8.3). MP-DCCP connections are closed by including an MP_CLOSE option in subflow DCCP-CloseReq or DCCP-Close messages. An MP-DCCP connection may also be reset through the use of an MP_FAST_CLOSE option. Key data from the initial handshake is included in the MP_CLOSE and MP_FAST_CLOSE to protect from unauthorized shutdown of MP-DCCP connections.

## 3. MP-DCCP Protocol

The DCCP protocol feature list (Section 6.4 of [RFC4340]) is extended in this document by adding a new Multipath feature with Feature number 10, as shown in Table 1.

| Number | Meaning | Rec'n Rule | Initial Value | Req'd |
|---|---|---|---|---|
| 10 | Multipath Capable | SP | 0 | N |

Table 1: Multipath feature

Rec'n Rule: The reconciliation rule used for the feature. SP indicates the server-priority as defined in section 6.3 of [RFC4340].

Initial Value: The initial value for the feature. Every feature has a known initial value.

Req'd: This column is "Y" if and only if every DCCP implementation MUST understand the feature. If it is "N", then the feature behaves like an extension, and it is safe to respond to Change options for the feature with empty Confirm options.

This specification adds a DCCP protocol option as defined in ([RFC4340], Section 5.8) providing a new Multipath related variable-length option with option type 46, as shown in Table 2.

| Type | Option Length | Meaning | DCCP-Data? |
|---|---|---|---|
| 46 | variable | Multipath | Y |

Table 2: Multipath option set

Note to the RFC Editor: The Feature Number and Option Type reflect the temporary assignment by IANA and must be verified once again.

## 3.1. Multipath Capable Feature

A DCCP endpoint negotiates the Multipath Capable Feature to determine whether multipath extensions can be enabled for a DCCP connection.

The Multipath Capable feature (MP_CAPABLE) has feature number 10 and follows the structure for features given in [RFC4340] Section 6. Beside the negotiation of the feature itself, also one or several values can be exchanged. The value field specified here for the Multipath Capable feature has a length of one-byte and can be repeated several times within the DCCP option for feature negotiation. This can be for example required to announce support of different versions of the protocol. For that, the leftmost four bits in Figure 3 specify the compatible version of the MP-DCCP implementation and MUST be set to 0 following this specification. The four bits following the Version field are unassigned in version 0 and MUST be set to zero by the sender and MUST be ignored by the receiver.

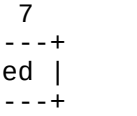



Figure 3: Format of the Multipath Capable feature value field

The setting of the MP_CAPABLE feature MUST follow the server-priority reconciliation rule described in ([RFC4340], Section 6.3.1). This allows multiple versions to be specified in order of priority.

The negotiation MUST be a part of the initial handshake procedure described in Section 3.3. No subsequent re-negotiation of the MP_CAPABLE feature is allowed for the same MP-DCCP connection.

Clients MUST include a Change R ([RFC4340], Section 6) option during the initial handshake request to supply a list of supported MP-DCCP protocol versions, ordered by preference.

Servers MUST include a Confirm L ([RFC4340], Section 6) option in the subsequent response to agree on an MP-DCCP version to be used from the Client list, followed by its own supported version(s), ordered by preference. Any subflow added to an existing MP-DCCP connection MUST use the version negotiated for the first subflow.

If no agreement is found, the Server MUST reply with an empty Confirm L option with feature number 10 and no values.

An example of successful version negotiation is shown hereafter and follows the negotiation example shown in [RFC4340] Section 6.5. For better understanding, this example uses the unspecified MP-DCCP versions 1 and 2 in addition to the MP-DCCP version 0 specified in this document:

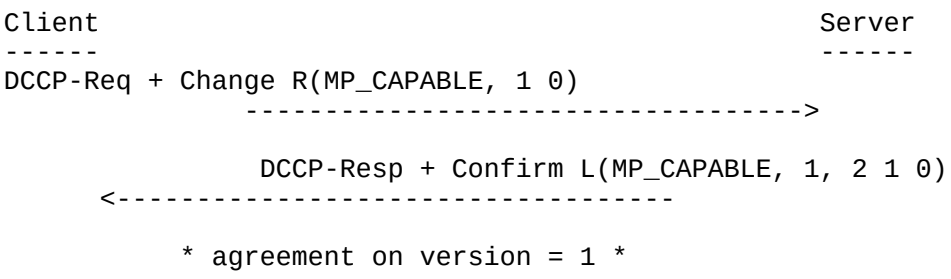



Figure 4: Example of MP-DCCP support negotiation using MP_CAPABLE

1. The Client indicates support for both MP-DCCP versions 1 and 0, with a preference for version 1.

2. Server agrees on using MP-DCCP version 1 indicated by the first value, and supplies its own preference list with the following values.

3. MP-DCCP is then enabled between the Client and Server with version 1.

Unlike the example in Figure 4, this document only allows the negotiation of MP-DCCP version 0. Therefore, successful negotiation of MP-DCCP as defined in this document, the client and the server MUST both support MP-DCCP version 0.

If the version negotiation fails or the MP_CAPABLE feature is not present in the DCCP-Request or DCCP-Response packets of the initial handshake procedure, the MP-DCCP connection MUST either fall back to regular DCCP or MUST close the connection. Further details are specified in Section 3.6

### 3.2. Multipath Option

MP-DCCP uses one single option to signal various multipath-related operations. The format of this multipath option is shown in Figure 5.

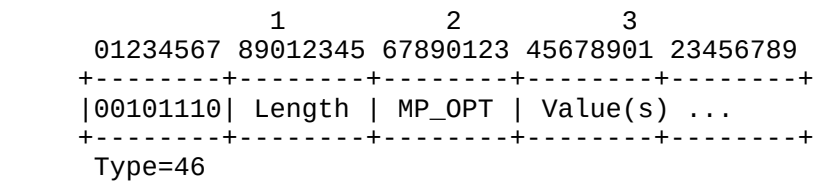



Figure 5: Multipath option format

The fields used by the multipath option are described in Table 3. MP_OPT refers to a Multipath option.

| Type | Option Length | MP_OPT | Meaning |
|---|---|---|---|
| 46 | var | 0 =MP_CONFIRM | Confirm reception and processing of an MP_OPT option |
| 46 | 12 | 1 =MP_JOIN | Join subflow to an existing MP-DCCP connection |
| 46 | var | 2 =MP_FAST_CLOSE | Close an MP-DCCP connection unconditionally |
| 46 | var | 3 =MP_KEY | Exchange key material for MP_HMAC |
| 46 | 9 | 4 =MP_SEQ | Multipath Sequence Number |
| 46 | 23 | 5 =MP_HMAC | Hash-based message auth. code for MP-DCCP |
| 46 | 12 | 6 =MP_RTT | Transmit RTT values and calculation parameters |
| 46 | var | 7 =MP_ADDADDR | Advertise additional address(es)/port(s) |
| 46 | 8 | 8 =MP_REMOVEADDR | Remove address(es)/ port(s) |
| 46 | 4 | 9 =MP_PRIO | Change subflow priority |
| 46 | var | 10 =MP_CLOSE | Close an MP-DCCP connection |
| 46 | var | 11 =MP_EXP | Experimental option for private use |
| 46 | TBD | >11 | Reserved for future Multipath options. |

Table 3: MP_OPT option types

Future MP options could be defined in a later version or extension to this specification.

These operations are largely inspired by the signals defined in [RFC8684]. The procedures for handling faulty or unknown MP options are described in Section 3.6.

### 3.2.1. MP_CONFIRM

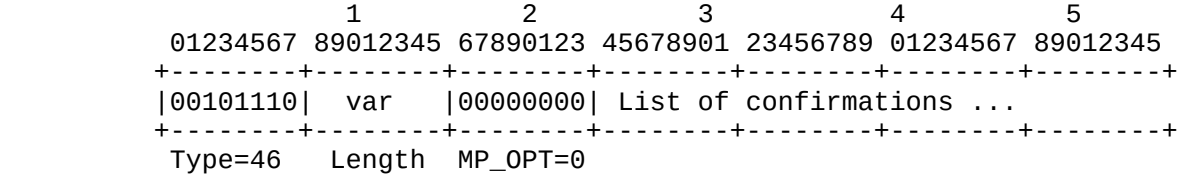



Figure 6: Format of the MP_CONFIRM option

Some multipath options require confirmation from the remote peer (see Table 4). Such options will be retransmitted by the sender until an MP_CONFIRM is received or the confirmation of options is considered irrelevant because the data contained in the options has already been replaced by newer information. This can happen, for example, with an MP_PRIO option if the path prioritization is changed while the previous prioritization has not yet been confirmed. The further processing of the multipath options in the receiving host is not the subject of MP_CONFIRM.

Multipath options could arrive out-of-order, therefore multipath options defined in Table 4 MUST be sent in a DCCP datagram with MP_SEQ; see Section 3.2.5. This allows a receiver to identify whether multipath options are associated with obsolete datasets (information carried in the option header) that would otherwise conflict with newer datasets. In the case of MP_ADDADDR or MP_REMOVEADDR the same dataset is identified based on AddressID, whereas the same dataset for MP_PRIO is identified by the subflow in use. An outdated multipath option is detected at the receiver if a previous multipath option referring to the same dataset contained a higher sequence number in the MP_SEQ. An MP_CONFIRM MAY be generated for multipath options that are identified as outdated.

Similarly, an MP_CONFIRM could arrive out of order. The associated MP_SEQ received MUST be echoed to ensure that the most recent multipath option is confirmed. This protects from inconsistencies that could occur, e.g. if three MP_PRIO options are sent one after the other on one path in order to first set the path priority to 0, then to 1 and finally to 0 again. Without an associated MP_SEQ, a loss of the third MP_PRIO option and a loss of the MP_CONFIRM of the second update and the third update would cause the sender to incorrectly interpret that the priority value was set to 0 without recognizing that the receiver has applied priority value 1.

The length of the MP_CONFIRM option and the path over which the option is sent depend on the confirmed multipath options and the received MP_SEQ, which are both copied verbatim and appended as a list of confirmations. The list is structured by first listing the received MP_SEQ followed by the related multipath option or options to confirm. The same rules apply when multipath options with different MP_SEQs are confirmed at once. This could happen if a datagram with MP_PRIO and a first MP_SEQ_1 and another datagram with MP_ADDADDR and a second MP_SEQ_2 are received in short succession. In this case, the structure described above is concatenated resulting in MP_SEQ_2 + MP_ADDADDR + MP_SEQ_1 + MP_PRIO. The order of the confirmed multipath options in the list of confirmations MUST reflect the incoming order at the host who sends the MP_CONFIRM, with the most recent suboption received listed first. This could allow the host receiving the MP_CONFIRM to verify that the options were applied in the correct order and to take countermeasures if they were not, e.g., if an MP_REMOVEADDR overtakes an MP_ADDADDR that refers to the same dataset.

| Type | Option Length | MP_OPT | MP_CONFIRM Sending path |
|---|---|---|---|
| 46 | var | 7 =MP_ADDADDR | Any available |
| 46 | 4 | 8 =MP_REMOVEADDR | Any available |
| 46 | 4 | 9 =MP_PRIO | Any available |

Table 4: Multipath options requiring confirmation

An example to illustrate the MP-DCCP confirm procedure for the MP_PRIO option is shown in Figure 7. The Host A sends a DCCP-Request on path A2-B2 with an MP_PRIO option with value 1 and associated sequence number of 1. Host B replies on the same path in this instance (any path can be used) with a DCCP-Response containing the MP_CONFIRM option and a list containing the original sequence number (1) together with the associated option (MP_PRIO).

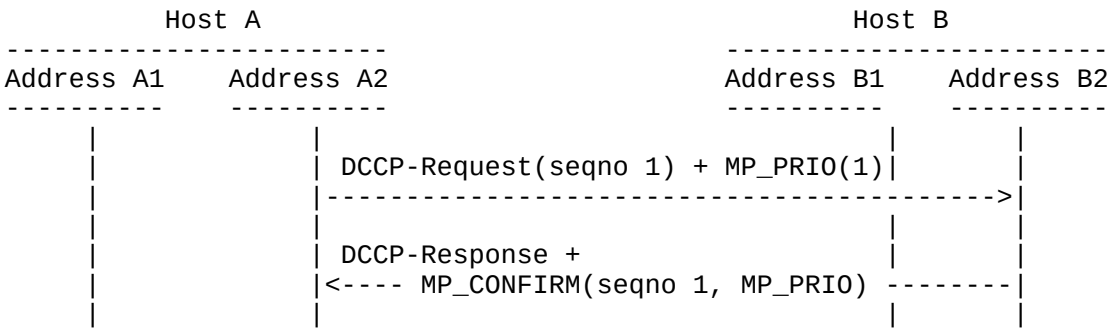



Figure 7: Example MP-DCCP CONFIRM procedure

A second example to illustrate the same MP-DCCP confirm procedure but where an out of date option is also delivered is shown in (Figure 8. Here, the first DCCP-Data is sent from Host A to Host B with option MP_PRIO set to 4. Host A subsequently sends the second DCCP-Data with option MP_PRIO set to 1. In this case, the delivery of the first MP_PRIO is delayed in the network between Host A and Host B and arrives after the second MP_PRIO. Host B ignores this second MP_PRIO as the associated sequence number is earlier than the first. Host B sends a DCCP-Ack confirming receipt of the MP_PRIO(1) with sequence number 2.

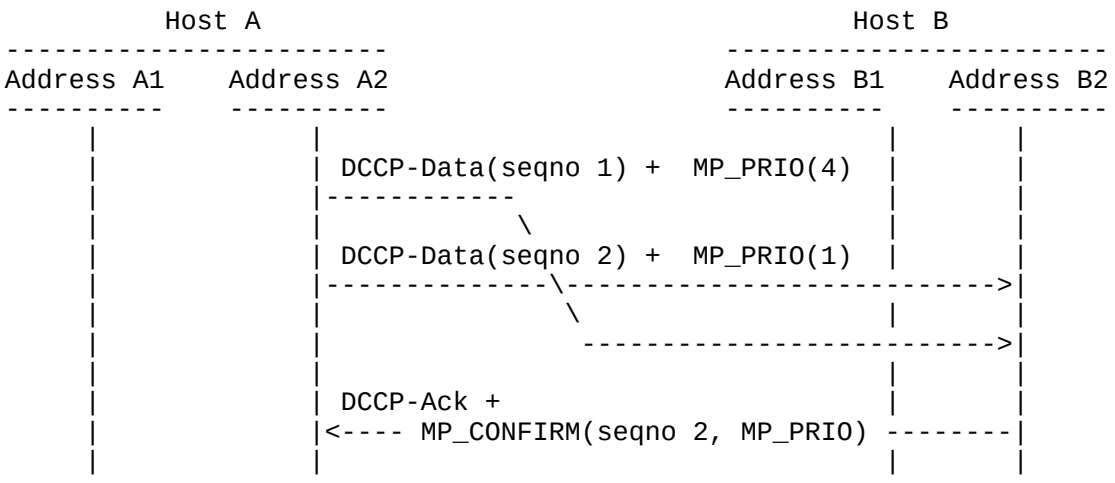



Figure 8: Example MP-DCCP CONFIRM procedure with outdated suboption

### 3.2.2. MP_JOIN

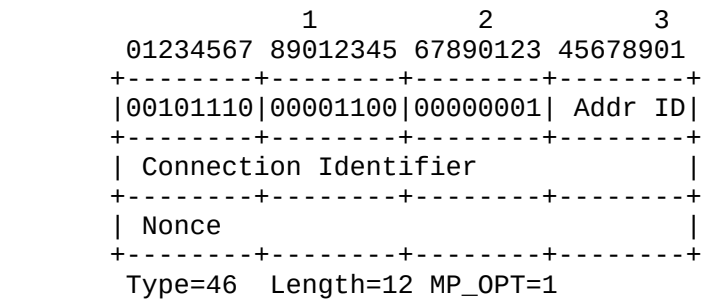



Figure 9: Format of the MP_JOIN suboption

The MP_JOIN option is used to add a new subflow to an existing MP-DCCP connection and REQUIRES a successful establishment of the first subflow using MP_KEY. The Connection Identifier (CI) is the one from the peer host, which was previously exchanged with the MP_KEY option. MP_HMAC MUST be set when using MP_JOIN within a DCCP-Response packet; see Section 3.2.6 for details. Similar to the setup of the first subflow, MP_JOIN also exchanges the Multipath Capable feature MP_CAPABLE as described in Section 3.1. This procedure includes the DCCP Confirm principle and thus ensures a reliable exchange of the MP_JOIN in accordance with section 6.6.4 of [RFC4340].

The MP_JOIN option includes an "Addr ID" (Address ID) generated by the sender of the option, used to identify the source address of this packet, even if the IP header was changed in transit by a middlebox. The value of this field is generated by the sender and MUST map uniquely to a source IP address for the sending host. The Address ID allows address removal (Section 3.2.9) without the need to know the source address at the receiver, thus allowing address removal through NATs. The Address ID also allows correlation between new subflow setup attempts and address signaling (Section 3.2.8), to prevent setting up duplicate subflows on the same path, if an MP_JOIN and MP_ADDADDR are sent at the same time.

The Address IDs of the subflow used in the initial DCCP Request/ Response exchange of the first subflow in the connection are implicit, and have the value zero. A host MUST store the mappings between Address IDs and addresses both for itself and the remote host. An implementation will also need to know which local and remote Address IDs are associated with which established subflows, for when addresses are removed from a local or remote host. An Address ID always MUST be unique over the lifetime of a subflow and can only be re-assigned if sender and receiver no longer have them in use.

The Nonce is a 32-bit random value locally generated for every MP_JOIN option. Together with the derived key from the both hosts Key Data described in Section 3.2.4, the Nonce value builds the basis to calculate the HMAC used in the handshaking process as described in Section 3.3 to avoid replay attacks.

If the CI cannot be verified by the receiving host during a handshake negotiation, the new subflow MUST be closed, as specified in Section 3.6.

### 3.2.3. MP_FAST_CLOSE

DCCP can send a Close or Reset signal to abruptly close a connection. Using MP-DCCP, a regular Close or Reset only has the scope of the subflow over which a signal was received. As such, it will only close the subflow and does not affect other remaining subflows or the MP-DCCP connection (unless it is the last subflow). This permits break-before-make handover between subflows.

In order to provide an MP-DCCP-level "reset" and thus allow the abrupt closure of the MP-DCCP connection, the MP_FAST_CLOSE suboption can be used.

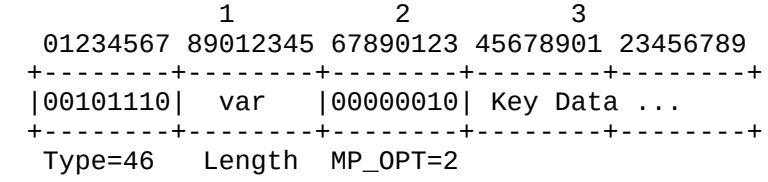



Figure 10: Format of the MP_FAST_CLOSE suboption

When Host A wants to abruptly close an MP-DCCP connection with Host B, it will send out the MP_FAST_CLOSE. The MP_FAST_CLOSE suboption MUST be sent from Host A on all subflows using a DCCP-Reset packet with Reset Code 13. The requirement to send the MP_FAST_CLOSE on all subflows increases the probability that Host B will receive the MP_FAST_CLOSE to take the same action. To protect from unauthorized shutdown of an MP-DCCP connection, the selected Key Data of the peer host during the handshaking procedure is carried by the MP_FAST_CLOSE option.

After sending the MP_FAST_CLOSE on all subflows, Host A MUST tear down all subflows and the multipath DCCP connection immediately terminates.

Upon reception of the first MP_FAST_CLOSE with successfully validated Key Data, Host B will send a DCCP-Reset packet response on all subflows to Host A with Reset Code 13 to clean potential middlebox states. Host B MUST then tear down all subflows and terminate the MP-DCCP connection.

### 3.2.4. MP_KEY

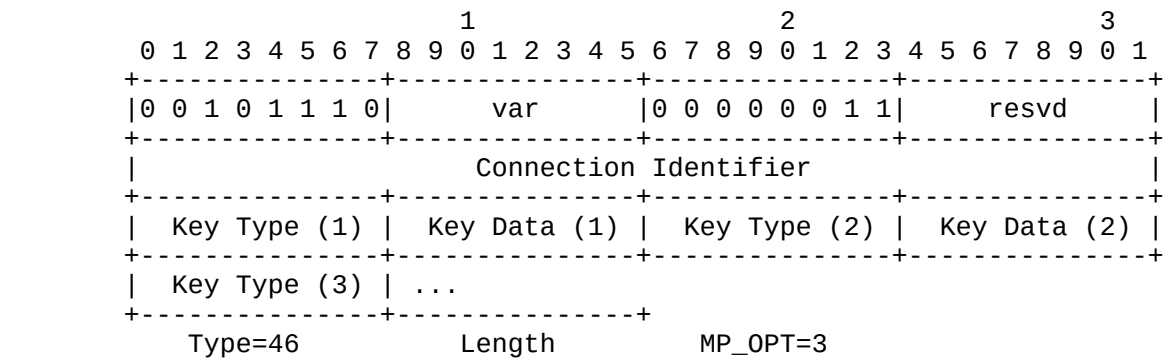



Figure 11: Format of the MP_KEY suboption

The MP_KEY suboption is used to exchange a Connection Identifier (CI) and key material between hosts (host A, host B) for a given connection. The CI is a unique number in the host for each multipath connection and is generated for inclusion in the first exchange of a connection with MP_KEY. With the CI it is possible to connect other DCCP subflows to an MP-DCCP connection with MP_JOIN (Section 3.2.2). Its size of 32-bits also defines the maximum number of simultaneous MP-DCCP connections in a host to 2^32. According to the Key related elements of the MP_KEY suboption, the Length varies between 17 and 73 bytes for a single-key message, and up to 82 bytes when all specified Key Types 0 and 255 are provided. The Key Type field specifies the type of the following key data. The set of key types are shown in Table 5.

| Key Type | Key Length (bytes) | Meaning |
|---|---|---|
| 0 =Plain Text | 8 | Plain Text Key |
| 1-254 | | Reserved for future Key Types |
| 255 =Experimental | 64 | For private use only |

Table 5: MP_KEY key types

Plain Text
: Key Data is exchanged in plain text between hosts (Host A, Host B), and the respective key parts (KeyA, KeyB) are used by each host to generate the derived key (d-key) by concatenating the two parts with the local key in front. That is,

  * Host A: d-keyA=(KeyA+KeyB)

  * Host B: d-keyB=(KeyB+KeyA)

Experimental
: This Key Type allows to use other Key Data and can be used to validate other key exchange mechanisms for a possible future specification.

Multiple keys are only permitted in the DCCP-Request message of the handshake procedure for the first subflow. This allows the hosts to agree on a single key type to be used, as described in Section 3.3

It is possible that not all hosts will support all key types and this specification does not recommend or enforce the announcement of any particular Key Type within MP_KEY option as this could have security implications. However, at least Key Type 0 (Plain Text) MUST be supported for interoperability tests in implementations of MP-DCCP. If the key type cannot be agreed in the handshake procedure, the MP-DCCP connection MUST fall back to not using MP-DCCP, as indicated in Section 3.6.

### 3.2.5. MP_SEQ

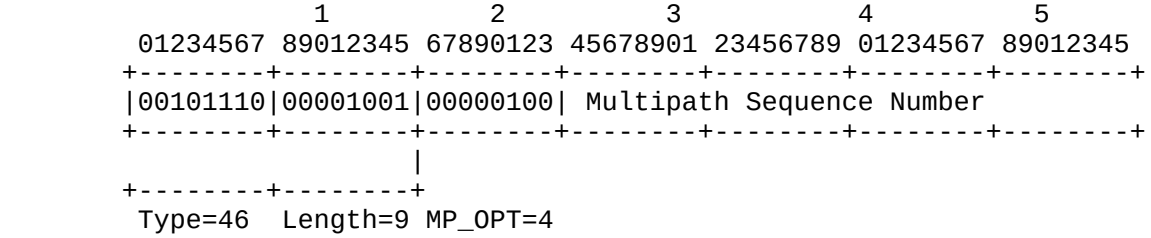



Figure 12: Format of the MP_SEQ suboption

The MP_SEQ suboption is used for end-to-end 48-bit datagram-based sequence numbers of an MP-DCCP connection. The initial data sequence number (IDSN) SHOULD be set randomly [RFC4086]. As with the standard DCCP sequence number, the data sequence number should not start at zero, but at a random value to make blind session hijacking more difficult, see also section 7.2 in [RFC4340].

The MP_SEQ number space is independent of the path individual sequence number space and MUST be sent with all DCCP-Data and DCCP-DataACK packets.

When the sequence number space is exhausted, the sequence number MUST be wrapped. [RFC7323] provides guidance on selecting an appropriately sized sequence number space according to the maximum segment lifetime of TCP. 64 bits is the recommended size for TCP to avoid the sequence number space going through within the segment lifetime. For DCCP, the Maximum Segment Lifetime is the same as that of TCP as specified in Section 3.4 of [RFC4340]. Compared to TCP, the sequence number for DCCP is incremented per packet rather than per byte transmitted. For this reason, the 48 bits chosen in MP_SEQ are considered sufficiently large considering the current globally routable maximum packet size of 1500 bytes, which corresponds to roughly 375 PiB of data within the sequence number space.

### 3.2.6. MP_HMAC

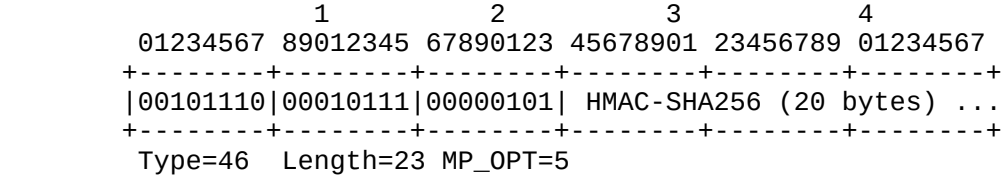



Figure 13: Format of the MP_HMAC suboption

The MP_HMAC suboption is used to provide authentication for the MP_ADDADDR, and MP_REMOVEADDR suboptions. In addition, it provides authentication for subflows joining an existing MP_DCCP connection, as described in the second and third step of the handshake of a subsequent subflow in Section 3.3. For this specification of MP-DCCP, the HMAC code is generated according to [RFC2104] in combination with the SHA256 hash algorithm described in [RFC6234], with the output in big-endian format truncated to the leftmost 160 bits (20 bytes). It is possible that other versions of MP-DCCP will define other hash algorithms in the future.

The "Key" used for the HMAC computation is the derived key (d-keyA for Host A or d-KeyB for Host B) described in Section 3.2.4, while the HMAC "Message" for MP_JOIN, MP_ADDADDR and MP_REMOVEADDR must be calculated in both hosts in order to protect the multipath option when sending and to validate the multipath option when receiving, and is a concatenation of:

* for MP_JOIN: The nonces of the MP_JOIN messages for which authentication shall be performed. Depending on whether Host A or Host B performs the HMAC-SHA256 calculation, it is carried out as follows:

  - MP_HMAC(A) = HMAC-SHA256(Key=d-keyA, Msg=RA+RB)

  - MP_HMAC(B) = HMAC-SHA256(Key=d-keyB, Msg=RB+RA)

A usage example is shown in Figure 21.

* for MP_ADDADDR: The Address ID and Nonce with associated IP address and if defined port, otherwise two bytes of value 0. IP address and port MUST be used in network byte order (NBO). Depending on whether Host A or Host B performs the HMAC-SHA256 calculation, it is carried out as follows:

  - MP_HMAC(A) = HMAC-SHA256(Key=d-keyA, Msg=Address ID+Nonce+NBO(IP)+NBO(Port))

  - MP_HMAC(B) = HMAC-SHA256(Key=d-keyB, Msg=Address ID+Nonce+NBO(IP)+NBO(Port))

* for MP_REMOVEADDR: Solely the Address ID. Depending on whether Host A or Host B performs the HMAC-SHA256 calculation, it is carried out as follows:

  - MP_HMAC(A) = HMAC-SHA256(Key=d-keyA, Msg=Address ID+Nonce)

  - MP_HMAC(B) = HMAC-SHA256(Key=d-keyB, Msg=Address ID+Nonce)

MP_JOIN, MP_ADDADDR and MP_REMOVEADDR can co-exist or be used multiple times within a single DCCP packet. All these multipath options require an individual MP_HMAC option. This ensures that the MP_HMAC is correctly associated. Otherwise, the receiver cannot validate multiple MP_JOIN, MP_ADDADDR or MP_REMOVEADDR. Therefore, an MP_HMAC MUST directly follow its associated multipath option. In the likely case of sending a MP_JOIN together with an MP_ADDADDR, this results in concatenating MP_JOIN + MP_HMAC_1 + MP_ADDADDR + MP_HMAC_2, whereas the first MP_HMAC_1 is associated with the MP_JOIN and the second MP_HMAC_2 is associated with the MP_ADDADDR suboption.

On the receiver side, the HMAC validation of the suboptions MUST be carried out according to the sending sequence in which the associated MP_HMAC follows a suboption. If the suboption cannot be validated by a receiving host because the HMAC validation fails (HMAC wrong or missing), the subsequent handling depends on which suboption was being verified. If the suboption to be authenticated was either MP_ADDADDR or MP_REMOVEADDR, the receiving host MUST silently ignore it (see Section 3.2.8 and Section 3.2.9). If the suboption to be authenticated was MP_JOIN, the subflow MUST be closed (see Section 3.6).

In the event that an MP_HMAC cannot be associated with a suboption this MP_HMAC MUST be ignored, unless it is a single MP_HMAC that was sent in a DCCP-Ack corresponding to a DCCP response packet with MP_JOIN (penultimate arrow in Figure 21).

### 3.2.7. MP_RTT

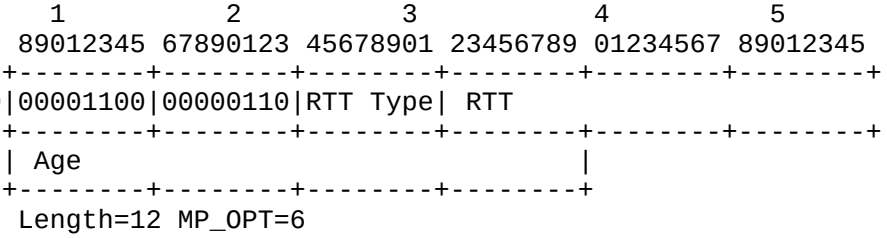



Figure 14: Format of the MP_RTT suboption

The MP_RTT suboption is used to transmit RTT values and age (represented in milliseconds) that belong to the path over which this information is transmitted. This information is useful for the receiving host to calculate the RTT difference between the subflows and to estimate whether missing data has been lost.

The RTT and Age information is a 32-bit integer. This covers a period of approximately 1193 hours.

The Field RTT type indicates the type of RTT estimation, according to the following description:

Raw RTT (=0)
: Raw RTT value of the last Datagram Round-Trip

Min RTT (=1)
: Min RTT value over a given period

Max RTT (=2)
: Max RTT value over a given period

Smooth RTT (=3)
: Averaged RTT value over a given period

Each CCID specifies the algorithms and period applied for their corresponding RTT estimations.The availability of the above described types, to be used in the MP_RTT option, depends on the CCID implementation in place.

Age
: The Age parameter defines the time difference between now - creation of the MP_RTT option - and the conducted RTT measurement in milliseconds. If no previous measurement exists, e.g., when initialized, the value is 0.

An example of a flow showing the exchange of path individual RTT information is provided in Figure 15. RTT1 refers to the first path and RTT2 to the second path. The RTT values could be extracted from the sender’s Congestion Control procedure and are conveyed to the receiving host using the MP_RTT suboption. With the reception of RTT1 and RTT2, the receiver is able to calculate the path_delta which corresponds to the absolute difference of both values. In the case that the path individual RTTs are symmetric in the down-link and up-link directions and there is no jitter, packets with missing sequence number MP_SEQ, e.g., in a reordering process, can be assumed lost after path_delta/2.

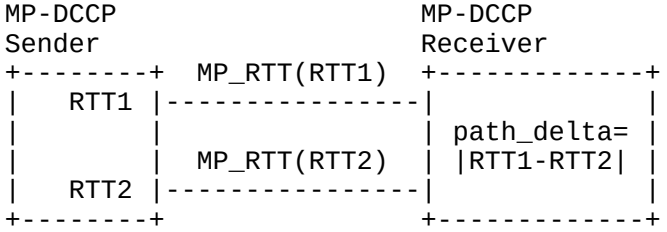



Figure 15: Exemplary flow of MP_RTT exchange and usage

### 3.2.8. MP_ADDADDR

The MP_ADDADDR suboption announces additional addresses (and, optionally, port numbers) by which a host can be reached. This can be sent at any time during an existing MP-DCCP connection, when the sender wishes to enable multiple paths and/or when additional paths become available. Multiple instances of this suboption within a packet can simultaneously advertise new addresses.

The Length is variable depending on the address family (IPv4 or IPv6) and whether a port number is used. This field is in range between 12 and 26 bytes.

The Nonce is a 32-bit random value that is generated locally for each MP_ADDADDR option and is used in the HMAC calculation process to prevent replay attacks.

The final 2 bytes, optionally specify the DCCP port number to use, and their presence can be inferred from the length of the option. Although it is expected that the majority of use cases will use the same port pairs as used for the initial subflow (e.g., port 80 remains port 80 on all subflows, as does the ephemeral port at the client), there could be cases (such as port-based load balancing) where the explicit specification of a different port is required. If no port is specified, the receiving host MUST assume that any attempt to connect to the specified address uses the port already used by the subflow on which the MP_ADDADDR signal was sent.

Along with the MP_ADDADDR option an MP_HMAC option MUST be sent for authentication. The truncated HMAC parameter present in this MP_HMAC option is the leftmost 20 bytes of an HMAC, negotiated and calculated as described in Section 3.2.6. In the same way as for MP_JOIN, the key for the HMAC algorithm, in the case of the message transmitted by Host A, will be d-KeyA, and in the case of Host B, d-KeyB. These are the keys that were exchanged and selected in the original MP_KEY handshake. The message for the HMAC is the Address ID, Nonce, IP address, and port number that precede the HMAC in the MP_ADDADDR option. If the port number is not present in the MP_ADDADDR option, the HMAC message will include 2 bytes of value zero. The rationale for the HMAC is to prevent unauthorized entities from injecting MP_ADDADDR signals in an attempt to hijack a connection. Note that, additionally, the presence of this HMAC prevents the address from being changed in flight unless the key is known by an intermediary. If a host receives an MP_ADDADDR option for which it cannot validate the HMAC, it MUST silently ignore the option.

The presence of an MP_SEQ (Section 3.2.5) MUST be ensured in a DCCP datagram in which MP_ADDADDR is sent, as described in Section 3.2.1.

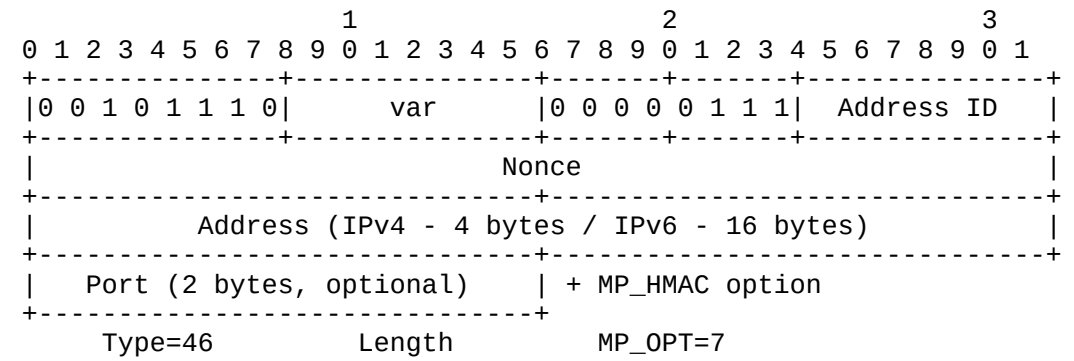



Figure 16: Format of the MP_ADDADDR suboption

Each address has an Address ID that could be used for uniquely identifying the address within a connection for address removal. Each host maintains a list of unique Address IDs and it manages these as it wishes. The Address ID is also used to identify MP_JOIN options (see Section 3.2.2) relating to the same address, even when address translators are in use. The Address ID MUST uniquely identify the address for the sender of the option (within the scope of the connection); the mechanism for allocating such IDs is implementation specific.

All Address IDs learned via either MP_JOIN or MP_ADDADDR can be stored by the receiver in a data structure that gathers all the Address-ID-to-address mappings for a connection (identified by a CI pair). In this way, there is a stored mapping between the Address ID, the observed source address, and the CI pair for future processing of control information for a connection. Note that an implementation MAY discard incoming address advertisements. Reasons for this are for example:

* to avoid the required mapping state, or

* because advertised addresses are of no use to it.

Possible scenarios in which this applies are the lack of resources to store a mapping or when IPv6 addresses are advertised even though the host only supports IPv4. Therefore, a host MUST treat address announcements as soft state. However, a sender MAY choose to update the announcements periodically to overcome temporary limitations.

A host MAY advertise private addresses, e.g., because there is a NAT on the path. It is desirable to allow this, since there could be cases where both hosts have additional interfaces on the same private network. The advertisement of broadcast or multicast IP addresses MUST be ignored by the recipient of this option, as it is not permitted according to the unicast principle of the basic DCCP.

The MP_JOIN handshake to create a new subflow (Section 3.2.2) provides mechanisms to minimize security risks. The MP_JOIN message contains a 32-bit CI that uniquely identifies a connection to the receiving host. If the CI is unknown, the host MUST send a DCCP-Reset.

Further security considerations around the issue of MP_ADDADDR messages that accidentally misdirect, or maliciously direct, new MP_JOIN attempts are discussed in Section 4. If a sending host of an MP_ADDADDR knows that no incoming subflows can be established at a particular address, an MP_ADDADDR MUST NOT announce that address unless the sending host has new knowledge about the possibility to do so. This information can be obtained from local firewall or routing settings, knowledge about availability of external NAT or firewall, or from connectivity checks performed by the host/application.

The reception of an MP_ADDADDR message is acknowledged using MP_CONFIRM (Section 3.2.1). This ensures reliable exchange of address information.

A host that receives an MP_ADDADDR, but finds at connection set up that the IP address and port number is unsuccessful, SHOULD NOT perform further connection attempts to this address/port combination for this connection to save resources. If a sender, however, wishes to trigger a new incoming connection attempt on a previously advertised address/port combination can therefore refresh the MP_ADDADDR information by sending the option again.

A host MAY send an MP_ADDADDR message with an already assigned Address ID using the IP Address previously assigned to this Address ID. The new MP_ADDADDR could have the same port number or a different port number. The receiver MUST silently ignore the MP_ADDADDR if the IP Address is not the same as that previously assigned to this Address ID. A host wishing to replace an existing Address ID MUST first remove the existing one (Section 3.2.9).

### 3.2.9. MP_REMOVEADDR

If, during the lifetime of an MP-DCCP connection, a previously announced address becomes invalid (e.g., if an interface disappears), the affected host SHOULD announce this. The peer can remove a previously added address with an Address ID from a connection using the Remove Address (MP_REMOVEADDR) suboption. This will terminate any subflows currently using that address.

MP_REMOVEADDR is only used to close already established subflows that have an invalid address. Functional flows with a valid address MUST be closed with a DCCP Close exchange (as with regular DCCP) instead of using MP_REMOVEADDR. For more information see Section 3.5.

The Nonce is a 32-bit random value that is generated locally for each MP_REMOVEADDR option and is used in the HMAC calculation process to prevent replay attacks.

Along with the MP_REMOVEADDR suboption a MP_HMAC option MUST be sent for authentication. The truncated HMAC parameter present in this MP_HMAC option is the leftmost 20 bytes of an HMAC, negotiated and calculated as described in Section 3.2.6. In the same way as for MP_JOIN, the key for the HMAC algorithm, in the case of the message transmitted by Host A, will be d-KeyA, and in the case of Host B, d-KeyB. These are the keys that were exchanged and selected in the original MP_KEY handshake. The message for the HMAC is the Address ID.

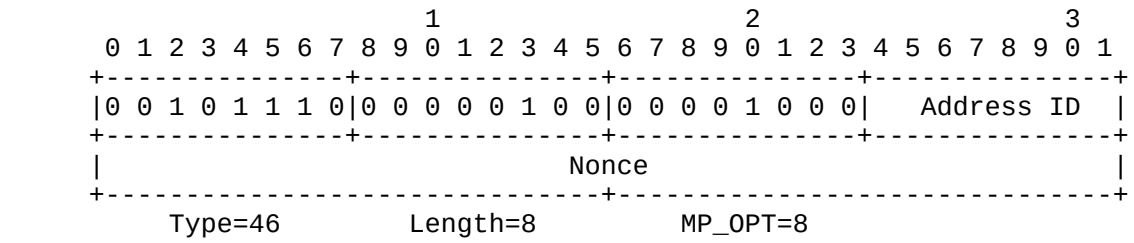
The rationale for using a HMAC is to prevent unauthorized entities from injecting MP_REMOVEADDR signals in an attempt to hijack a connection. Note that, additionally, the presence of this HMAC prevents the address from being modified in flight unless the key is known by an intermediary. If a host receives an MP_REMOVEADDR option for which it cannot validate the HMAC, it MUST silently ignore the option.

A receiver MUST include an MP_SEQ (Section 3.2.5) in a DCCP datagram that sends an MP_REMOVEADDR. Further details are given in Section 3.2.1.

The reception of an MP_REMOVEADDR message is acknowledged using MP_CONFIRM (Section 3.2.1). This ensures reliable exchange of address information. To avoid inconsistent states, the sender releases the address ID only after MP_REMOVEADDR has been confirmed.

The sending and receiving of this message SHOULD trigger the closing procedure described in [RFC4340] between the client and the server on the affected subflow(s), if possible. This helps remove middlebox state, before removing any local state.

Address removal is done by Address ID to allow the use of NATs and other middleboxes that rewrite source addresses. If there is no address at the requested Address ID, the receiver will silently ignore the request.




-> followed by MP_HMAC option

Figure 17: Format of the MP_REMOVEADDR suboption

### 3.2.10. MP_PRIO

The path priority signaled with the MP_PRIO option provides hints for the packet scheduler when making decisions about which path to use for payload traffic. When a single specific path from the set of available paths is treated with higher priority compared to the others when making scheduling decisions for payload traffic, a host can signal such change in priority to the peer. This could be used when there are different costs for using different paths (e.g., Wi-Fi is free while cellular has limit on volume, 5G has higher energy consumption). The priority of a path could also change, for example, when a mobile host runs out of battery, the usage of only a single path may be the preferred choice of the user.

The MP_PRIO suboption, shown below, can be used to set a priority value for the subflow over which the suboption is received.

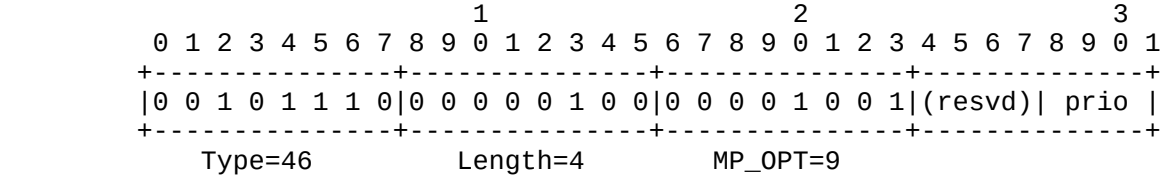



Figure 18: Format of the MP_PRIO suboption

The following values are available for the Prio field:

* 0: Do not use. The path is not available.

* 1: Standby: do not use this path for traffic scheduling, if another path (secondary or primary) is available. The path will only be used if other secondary or primary paths are not established.

* 2: Secondary: do not use this path for traffic scheduling, if the other paths are good enough. The path will be used occasionally for increasing temporarily the available capacity, e.g. when primary paths are congested or are not available. This is the recommended setting for paths that have costs or data caps as these paths will be used less frequently then primary paths.

* 3 - 15: Primary: The path can be used for packet scheduling decisions. The priority number indicates the relative priority of one path over the other for primary paths. Higher numbers indicate higher priority. The peer should consider sending traffic first over higher priority paths. This is the recommended setting for paths that do not have a cost or data caps associated with them as these paths will be frequently used.

Example use cases include:

1. Setting Wi-Fi path to Primary and Cellular paths to Secondary. In this case Wi-Fi will be used and Cellular will be used only if the Wi-Fi path is congested or not available. Such setting results in using the Cellular path only temporally, if more capacity is needed than the Wi-Fi path can provide, indicating a clear priority of the Wi-Fi path over the Cellular due to, e.g., cost reasons.

2. Setting Wi-Fi path to Primary and Cellular to Standby. In this case Wi-Fi will be used and Cellular will be used only if the Wi-Fi path is not available.

3. Setting Wi-Fi path to Primary and Cellular path to Primary. In this case, both paths can be used when making packet scheduling decisions.

If not specified, the default behavior is to always use a path for packet scheduling decisions (MP_PRIO=3), when the path has been established and added to an existing MP-DCCP connection. At least one path ought to have an MP_PRIO value greater or equal to one for it to be allowed to send on the connection. It is RECOMMENDED to update at least one path to a non-zero MP_PRIO value when an MP-DCCP

connection enters a state where all paths remain with an MP_PRIO value of zero. This helps an MP-DCCP connection to schedule when the multipath scheduler strictly respects MP_PRIO value 0. To ensure reliable transmission, the MP_PRIO suboption MUST be acknowledged via an MP_CONFIRM (see Table 4).

The relative ratio of the primary path values 3-15 depend on the path usage strategy, which is described in more detail in Section 3.11. In the case of path mobility (Section 3.11.1), only one path can be used at a time and MUST be the appropriate one that has the highest available priority value including also the prio numbers 1 and 2. In the other case of concurrent path usage (Section 3.11.2), the definition is up to the multipath scheduler logic.

An MP_SEQ (Section 3.2.5) MUST be present in a DCCP datagram in which the MP_PRIO suboption is sent. Further details are given in Section 3.2.1.

### 3.2.11. MP_CLOSE

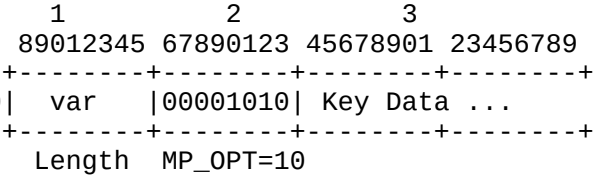



Figure 19: Format of the MP_CLOSE suboption

An MP-DCCP connection can be gracefully closed by sending and MP_CLOSE to the peer host. On all subflows, the regular termination procedure as described in [RFC4340] MUST be initiated using MP_CLOSE in the initial packet (either a DCCP-CloseReq or a DCCP-Close). When a DCCP-CloseReq is used, the following DCCP-Close MUST also carry the MP_CLOSE to avoid keeping a state in the sender of the DCCP-CloseReq. At the initiator of the DCCP-CloseReq, all sockets including the MP-DCCP connection socket, transition to CLOSEREQ state. To protect from unauthorized shutdown of a multi-path connection, the selected Key Data of the peer host during the handshaking procedure MUST be included in by the MP_CLOSE option and must be validated by the peer host. Note, the Key Data is different between MP_CLOSE option carried by DCCP-CloseReq or DCCP-Close.

On reception of the first DCCP-CloseReq carrying an MP_CLOSE with valid Key Data, or due to a local decision, all subflows transition to the CLOSING state before transmitting a DCCP-Close carrying MP_CLOSE. The MP-DCCP connection socket on the host sending the DCCP-Close reflects the state of the initial subflow during handshake with MP_KEY option. If the initial subflow no longer exists, the state moves immediately to CLOSED.

Upon reception of the first DCCP-Close carrying an MP_CLOSE with valid Key Data at the peer host, all subflows, as well as the MP-DCCP connection socket, move to the CLOSED state. After this, a DCCP-Reset with Reset Code 1 MUST be sent on any subflow in response to a received DCCP-Close containing a valid MP_CLOSE option.

When the MP-DCCP connection socket is in CLOSEREQ or CLOSE state, new subflow requests using MP_JOIN MUST be ignored.

Contrary to an MP_FAST_CLOSE (Section 3.2.3), no single-sided abrupt termination is applied.

### 3.2.12. Experimental Multipath option MP_EXP for private use

This section reserves a Multipath option to define and specify any experimental additional feature for improving and optimization of the MP-DCCP protocol. This option could be applicable to specific environments or scenarios according to potential new requirements and is meant for private use only. MP_OPT feature number 11 is specified with an exemplary description as below:

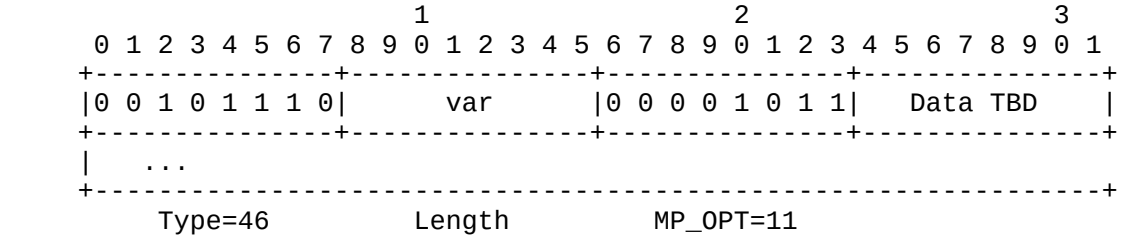



Figure 20: Format of the MP_EXP suboption

The Data field can carry any data according to the foreseen use by the experimenters with a maximum length of 252 bytes.

## 3.3. MP-DCCP Handshaking Procedure

An example to illustrate the MP-DCCP handshake procedure is shown in Figure 21.

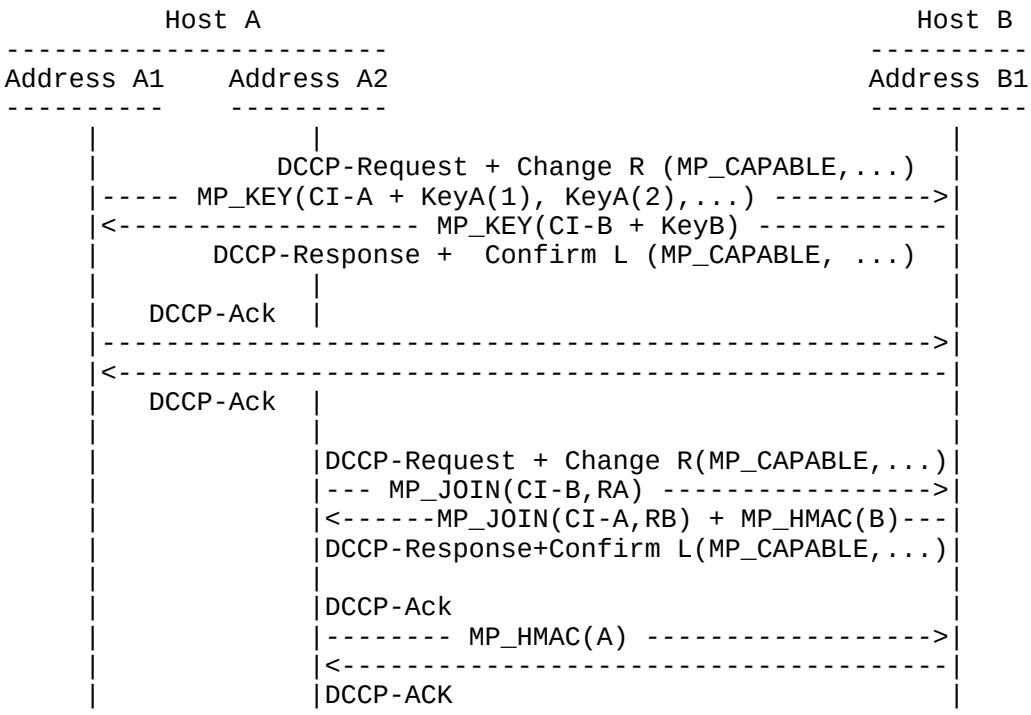



Figure 21: Example MP-DCCP handshake

The basic initial handshake for the first subflow is as follows:

* Host A sends a DCCP-Request with the MP-Capable feature Change request and the MP_KEY option with a Host-specific CI-A and a KeyA for each of the supported key types as described in Section 3.2.4. CI-A is a unique identifier during the lifetime of an MP-DCCP connection.

* Host B sends a DCCP-Response with Confirm feature for MP-Capable and the MP_Key option with a unique Host-specific CI-B and a single Host-specific KeyB. The type of the key is chosen from the list of supported types from the previous request.

* Host A sends a DCCP-Ack to confirm the proper key exchange.

* Host B sends a DCCP-Ack to complete the handshake and set both connection ends to the OPEN state.

It should be noted that DCCP is protected against corruption of DCCP header data (section 9 of [RFC4340]), so no additional mechanisms beyond the general confirmation are required to ensure that the header data has been properly received.

Host A waits for the final DCCP-Ack from Host B before starting any establishment of additional subflow connections.

The handshake for subsequent subflows based on a successful initial handshake is as follows:

* Host A sends a DCCP-Request with the MP-Capable feature Change request and the MP_JOIN option with Host B's CI-B, obtained during the initial handshake. Additionally, an own random nonce RA is transmitted with the MP_JOIN.

* Host B computes the HMAC of the DCCP-Request and sends a DCCP-Response with Confirm feature option for MP-Capable and the MP_JOIN option with the CI-A and a random nonce RB together with the computed MP_HMAC. As specified in Section 3.2.6, the HMAC is calculated by taking the leftmost 20 bytes from the SHA256 hash of a HMAC code created by using the nonce received with MP_JOIN(A) and the local nonce RB as message and the derived key described in Section 3.2.4 as key:

  MP_HMAC(B) = HMAC-SHA256(Key=d-keyB, Msg=RB+RA)

* Host A sends a DCCP-Ack with the HMAC computed for the DCCP-Response. As specified in Section 3.2.6, the HMAC is calculated by taking the leftmost 20 bytes from the SHA256 hash of a HMAC code created by using the local nonce RA and the nonce received with MP_JOIN(B) as message and the derived key described in Section 3.2.4 as key:

  MP_HMAC(A) = HMAC-SHA256(Key=d-keyA, Msg=RA+RB)

* Host B sends a DCCP-Ack to confirm the HMAC and to conclude the handshaking.

## 3.4. Address knowledge exchange

### 3.4.1. Advertising a new path (MP_ADDADDR)

When a host (Host A) wants to advertise the availability of a new path, it should use the MP_ADDADDR option (Section 3.2.8) as shown in the example in Figure 22. The MP_ADDADDR option passed in the DCCP-Data contains the following parameters:

* an identifier (id 2) for the new IP address which is used as a reference in subsequent control exchanges.

* a Nonce value to prevent replay attacks

* the IP address of the new path (A2_IP)

* A pair of bytes specifying the port number associated with this IP address. The value of 00 here indicates that the port number is the same as that used for the initial subflow address A1_IP

According to Section 3.2.8, the following options are required in a packet carrying MP_ADDADDR:

* the leftmost 20 bytes of the HMAC(A) generated during the initial handshaking procedure described in Section 3.3 and Section 3.2.6

* the MP_SEQ option with the sequence number (seqno 12) for this message according to Section 3.2.5.

Host B acknowledges receipt of the MP_ADDADDR message with a DCCP-Ack containing the MP_CONFIRM option. The parameters supplied in this response are as follows:

* an MP_CONFIRM containing the MP_SEQ number (seqno 12) of the packet carrying the option that we are confirming together with the MP_ADDADDR option

* the leftmost 20 bytes of the HMAC(B) generated during the initial handshaking procedure Section 3.3

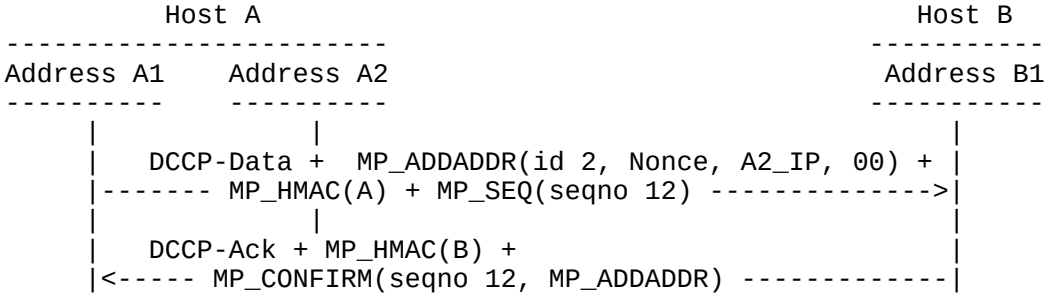



Figure 22: Example MP-DCCP ADDADDR procedure

### 3.4.2. Removing a path (MP_REMOVEADDR)

When a host (Host A) wants to indicate that a path is no longer available, it should use the MP_REMOVEADDR option (Section 3.2.9) as shown in the example in Figure 23. The MP_REMOVEADDR option passed in the DCCP-Data contains the following parameters:

* an identifier (id 2) for the IP address to remove (A2_IP) and which was specified in a previous MP_ADDADDR message.

* a Nonce value to prevent replay attacks

According to Section 3.2.9, the following options are required in a packet carrying MP_REMOVEADDR:

* the leftmost 20 bytes of the HMAC(A) generated during the initial handshaking procedure described in Section 3.3 and Section 3.2.6

* the MP_SEQ option with the sequence number (seqno 33) for this message according to Section 3.2.5.

Host B acknowledges receipt of the MP_REMOVEADDR message with a DCCP-Ack containing the MP_CONFIRM option. The parameters supplied in this response are as follows:

* an MP_CONFIRM containing the MP_SEQ number (seqno 33) of the packet carrying the option that we are confirming, together with the MP_REMOVEADDR option

* the leftmost 20 bytes of the HMAC(B) generated during the initial handshaking procedure Section 3.3

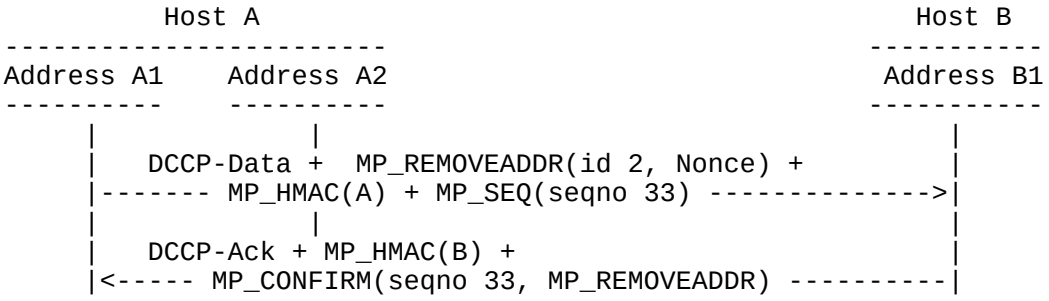



Figure 23: Example MP-DCCP REMOVEADDR procedure

### 3.5. Closing an MP-DCCP connection

When a host wants to close an existing subflow but not the whole MP-DCCP connection, it MUST initiate the regular DCCP connection termination procedure as described in Section 5.6 of [RFC4340], i.e., it sends a DCCP-Close/DCCP-Reset on the subflow. This may be preceded by a DCCP-CloseReq. In the event of an irregular termination of a subflow, e.g., during subflow establishment, it MUST use an appropriate DCCP-Reset code as specified in IANA [DCCP.Parameter] for DCCP operations. This could be, for example, sending reset code 5 (Option Error) when an MP-DCCP option provides invalid data or reset code 9 (Too Busy) when the maximum number of maintainable paths is reached. Note that receiving a reset code 9 for secondary subflows MUST NOT impact already existing active subflows. If necessary, these subflows are terminated in a subsequent step using the procedures described in this section.

A host terminates an MP-DCCP connection using the DCCP connection termination specified in section 5.5 of [RFC4340] on each subflow with the first packet on each subflow carrying MP_CLOSE (see Section 3.2.11).

```
  Host A                                  Host B
  ------                                  ------
                                  <-      Optional DCCP-CloseReq +
                                          MP_CLOSE [A's key]
                                          [on all subflows]
  DCCP-Close + MP_CLOSE           ->
  [B's key] [on all subflows]
                                  <-      DCCP-Reset
                                          [on all subflows]
```

Additionally, an MP-DCCP connection may be closed abruptly using the "Fast Close" procedure described in Section 3.2.3, where a DCCP-Reset is sent on all subflows, each carrying the MP_FAST_CLOSE option.

```
  Host A                                  Host B
  ------                                  ------
  DCCP-Reset + MP_FAST_CLOSE      ->
  [B's key] [on all subflows]
                                  <-      DCCP-Reset
                                          [on all subflows]
```

### 3.6. Fallback

When a subflow fails to operate following MP-DCCP intended behavior, it is necessary to proceed with a fall back. This may be either falling back to regular DCCP [RFC4340] or removing a problematic subflow. The main reasons for subflow failing include: no MP support at peer host, failure to negotiate protocol version, loss of Multipath options, faulty/non-supported MP-DCCP options or modification of payload data.

At the start of an MP-DCCP connection, the handshake ensures exchange of MP-DCCP feature and options and thus ensures that the path is fully MP-DCCP capable. If during the handshake procedure it appears that DCCP-Request or DCCP-Response messages do not carry the MP_CAPABLE feature, the MP-DCCP connection will not be established and the handshake SHOULD fall back to regular DCCP. If this is not possible the connection MUST be closed.

If the endpoints fail to agree on the protocol version to use during the Multipath Capable feature negotiation, the connection MUST either be closed or fall back to regular DCCP. This is described in Section 3.1. The protocol version negotiation distinguishes between negotiation for the initial connection establishment, and addition of subsequent subflows. If protocol version negotiation is not successful during the initial connection establishment, MP-DCCP connection will fall back to regular DCCP.

The fall back procedure to regular DCCP MUST be also applied if the MP_KEY Section 3.2.4 Key Type cannot be negotiated.

If a subflow attempts to join an existing MP-DCCP connection, but MP-DCCP options or MP_CAPABLE feature are not present or are faulty in the handshake procedure, that subflow MUST be closed. This is especially the case if a different MP_CAPABLE version than the originally negotiated version is used. Reception of a non-verifiable MP_HMAC (Section 3.2.6) or an invalid CI used in MP_JOIN (Section 3.2.2) during flow establishment MUST cause the subflow to be closed.

The subflow closing procedure MUST be also applied if a final ACK carrying MP_KEY with wrong KeyA/KeyB is received or MP_KEY option is malformed.

Another relevant case is when payload data is modified by middleboxes. DCCP uses checksum to protect the data, as described in section 9 of [RFC4340]. A checksum will fail if the data has been changed in any way. All data from the start of the segment that failed the checksum onwards cannot be considered trustworthy. DCCP

defines that if the checksum fails, the receiving endpoint MUST drop the application data and report that data as dropped due to corruption using a Data Dropped option (Drop Code 3, Corrupt). If data is dropped due to corruption for an MP-DCCP connection, the affected subflow MAY be closed. The same procedure applies if the MP option is unknown.

### 3.7. State Diagram

The MP-DCCP per subflow state transitions to a large extent follow the state transitions defined for DCCP in [RFC4340], with some modifications due to the MP-DCCP four-way handshake and fast close procedures. The state diagram below illustrates the most common state transitions. The diagram is illustrative. For example, there are arcs (not shown) from several additional states to TIMEWAIT, contingent on the receipt of a valid DCCP-Reset.

The states transitioned when moving from the CLOSED to OPEN state during the four-way handshake remain the same as for DCCP, but it is no longer possible to transmit application data while in the REQUEST state. The fast close procedure can be triggered by either the client or the server and results in the transmission of a Reset packet. The fast close procedure moves the state of the client and server directly to TIMEWAIT and CLOSED, respectively.

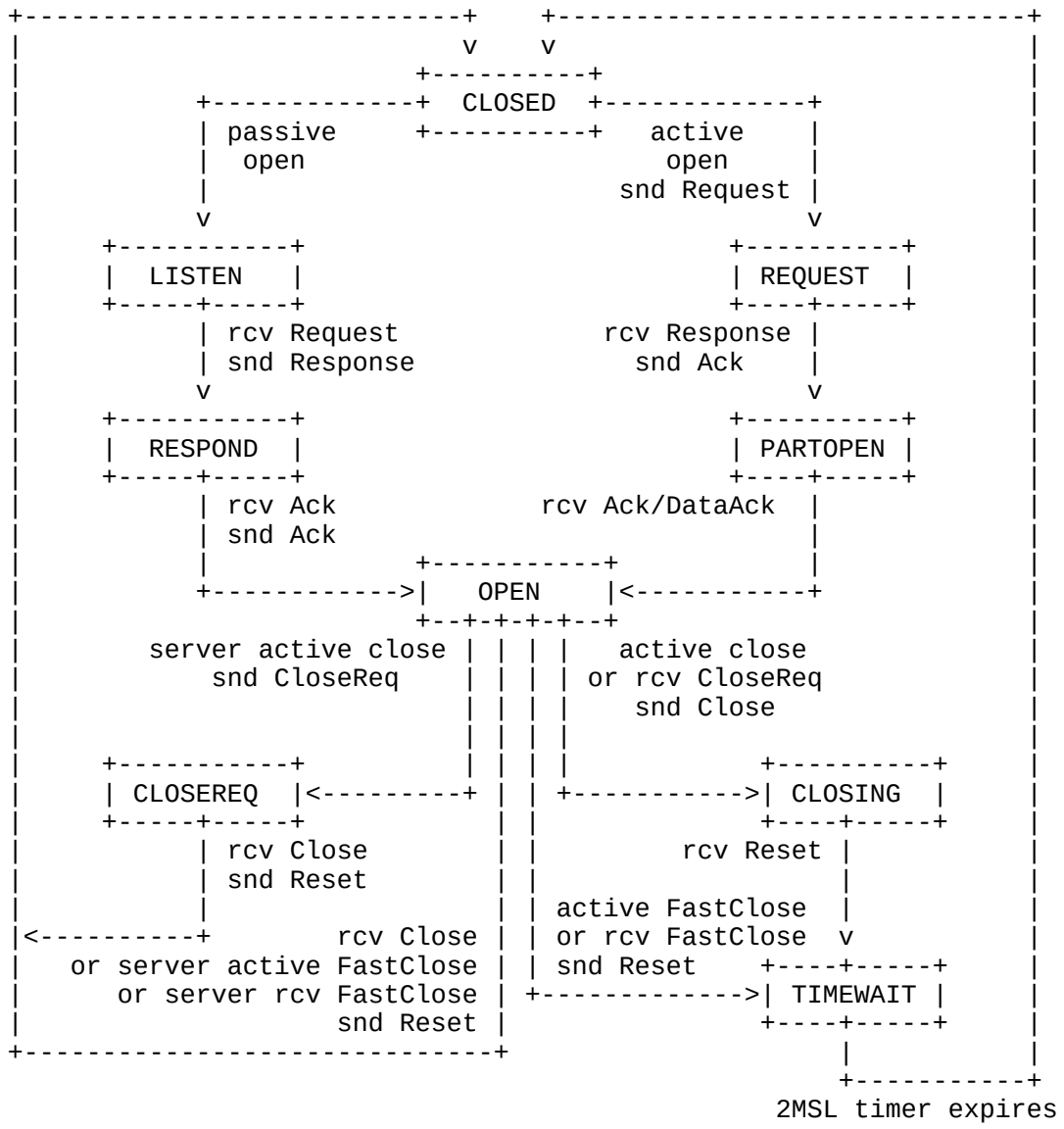



Figure 24: Most common state transitions of an MP-DCCP subflow

## 3.8. Congestion Control Considerations

Senders MUST manage per-path congestion status, and avoid to sending more data on a given path than congestion control for each path allows.

### 3.9. Maximum Packet Size Considerations

A DCCP implementation maintains the maximum packet size (MPS) during operation of a DCCP session. This procedure is specified for single-path DCCP in [RFC4340], Section 14. Without any restrictions, this is adopted for MP-DCCP operations, in particular the PMTU measurement and the Sender Behaviour. The DCCP application interface SHOULD allow the application to discover the current MPS. This reflects the current supported largest size for the data stream that can be used across the set of all active MP-DCCP subflows.

### 3.10. Maximum number of Subflows Considerations

MP-DCCP does not support any explicit procedure to negotiate the maximum number of subflows between endpoints. In practical scenarios, however, there will be resource limitations on the host or use cases that do not benefit from additional subflows.

It is RECOMMENDED to limit the number of subflows in implementations and to reject incoming subflow requests with a DCCP-Reset using the Reset Code "too busy" according to [RFC4340] if the resource limit is exceeded or it is known that the multipath connection will not benefit from further subflows. Likewise, the host that wants to create the subflows is RECOMMENDED to consider the aspect of available resources and the possible gains.

To avoid further inefficiencies with subflows due to short-lived connections, it MAY be useful to delay the start of additional subflows. The decision on the initial number of subflows can be based on the occupancy of the socket buffer and/or the timing.

While in the socket buffer based approach the number of initial subflows can be derived by opening new subflows until their initial windows cover the amount of buffered application data, the timing based approach delays the start of additional subflows based on a certain time period, load or knowledge of traffic and path properties. The delay based approach also provides resilience for low-bandwidth but long-lived applications. All this could also be supported by advanced APIs that signal application traffic requests to the MP-DCCP.

### 3.11. Path usage strategies

MP-DCCP can be configured to realize one of several strategies for path usage, via selecting one DCCP subflow of the multiple DCCP subflows within an MP-DCCP connection for data transmission. This can be a dynamic process further facilitated by the means of DCCP and MP-DCCP defined options such as path preference using MP-PRIO, adding or removing DCCP subflows using MP_REMOVEADDR, MP_ADDADDR or DCCP-Close/DCCP-Reset and also path metrics such as packet-loss-rate, CWND or RTT provided by the Congestion Control Algorithm. Selecting an appropriate method can allow MP-DCCP to realize different path utilization strategies that make MP-DCCP suitable for end-to-end implementation over the Internet or in controlled environments such as Hybrid Access or 5G ATSSS.

#### 3.11.1. Path mobility

The path mobility strategy provides the use of a single path with a seamless handover function to continue the connection when the currently used path is deemed unsuitable for service delivery. Some of the DCCP subflows of an MP-DCCP connection might become inactive due to either the occurrence of certain error conditions (e.g., DCCP timeout, packet loss threshold, RTT threshold, closed/removed) or adjustments from the MP-DCCP user. When there is outbound data to send and the primary path becomes inactive (e.g., due to failures) or de-prioritized, the MP-DCCP endpoint SHOULD try to send the data through an alternate path with a different source or destination address (depending on the point of failure), if one exists. This process SHOULD respect the path priority configured by the MP_PRIO suboption or if not available pick the most divergent source-destination pair from the original used source-destination pair.

> Note: Rules for picking the most appropriate source-destination pair are an implementation decision and are not specified within this document. Path mobility is supported in the current Linux reference implementation [multipath-dccp.org].

#### 3.11.2. Concurrent path usage

Different to a path mobility strategy, the selection between MP-DCCP subflows is a per-packet decision that is a part of the multipath scheduling process. This method would allow multiple subflows to be simultaneously used to aggregate the path resources to obtain higher connection throughput.
In this scenario, the selection of congestion control, per-packet scheduling and potential re-ordering method determines a concurrent path utilization strategy and result in a particular transport characteristic. A concurrent path usage method uses a scheduling

design that could seek to maximize reliability, throughput, minimizing latency, etc.

Concurrent path usage over the Internet can have implications. When a Multipath DCCP connection uses two or more paths, there is no guarantee that these paths are fully disjoint. When two (or more) subflows share the same bottleneck, using a standard congestion control scheme could result in an unfair distribution of the capacity with the multipath connection using more capacity than competing single path connections.
Multipath TCP uses the coupled congestion control Linked Increases Algorithm (LIA) specified in the experimental specification [RFC6356] to solve this problem. This scheme could also be specified for Multipath DCCP. The same applies to other coupled congestion control schemes that have been proposed for Multipath TCP such as Opportunistic Linked Increases Algorithm [OLIA].

The specification of scheduling for concurrent multipath and related the congestion control algorithms and re-ordering methods for use in the general Internet are outside the scope of this document. If, and when, the IETF specifies a method for concurrent usage of multiple paths for the general Internet, the framework specified in this document could be used to provide an IETF recommended method for MP-DCCP.

## 4. Security Considerations

Similar to DCCP, MP-DCCP does not provide cryptographic security guarantees inherently. Thus, if applications need cryptographic security (integrity, authentication, confidentiality, access control, and anti-replay protection) the use of IPsec, DTLS over DCCP [RFC5238] or other end-to-end security is recommended; Secure Real-time Transport Protocol (SRTP) [RFC3711] is one candidate protocol for authentication. Together with Encryption of Header Extensions in SRTP, as provided by [RFC6904], also integrity would be provided.

DCCP [RFC4340] provides protection against hijacking and limits the potential impact of some denial-of-service attacks, but DCCP provides no inherent protection against an on-path attacker snooping on data packets. Regarding the security of MP-DCCP no additional risks should be introduced compared to regular DCCP. The security objectives for MP-DCCP are:

* Provide assurance that the parties involved in an MP-DCCP handshake procedure are identical to those in the original DCCP connection.

* Before a path is used, verify that the new advertised path is valid for receiving traffic.

* Provide replay protection, i.e., ensure that a request to add/remove a subflow is 'fresh'.

* Allow a party to limit the number of subflows that it allows.

To achieve these goals, MP-DCCP includes a hash-based handshake algorithm documented in Sections Section 3.2.4, Section 3.2.6 and Section 3.3. The security of the MP-DCCP connection depends on the use of keys that are shared once at the start of the first subflow and are never sent again over the network. Depending on the security requirements, different Key Types can be negotiated in the handshake procedure or must follow the fallback scenario described in Section 4. If there are security requirements that go beyond the capabilities of Key Type 0, then it is RECOMMENDED that Key Type 0 is not enabled to avoid downgrade attacks that result in the key being exchanged as plain text. To ease demultiplexing while not revealing cryptographic material, subsequent subflows use the initially exchanged CI information. The keys exchanged once at the beginning are concatenated and used as keys for creating Hash-based Message Authentication Codes (HMACs) used on subflow setup, in order to verify that the parties in the handshake of subsequent subflows are the same as in the original connection setup. This also provides verification that the peer can receive traffic at this new address. Replay attacks would still be possible when only keys are used; therefore, the handshakes use single-use random numbers (nonces) for both parties -- this ensures that the HMAC will never be the same on two handshakes. Guidance on generating random numbers suitable for use as keys is given in [RFC4086]. During normal operation, regular DCCP protection mechanisms (such as header checksum to protect DCCP headers against corruption) is designed to provide the same level of protection against attacks on individual DCCP subflows as exists for regular DCCP.

As discussed in Section 3.2.8, a host may advertise its private addresses, but these might point to different hosts in the receiver's network. The MP_JOIN handshake (Section 3.2.2) is designed to ensure that this does not set up a subflow to the incorrect host. However, it could still create unwanted DCCP handshake traffic. This feature of MP-DCCP could be a target for denial-of-service exploits, with malicious participants in MP-DCCP connections encouraging the recipient to target other hosts in the network. Therefore, implementations should consider heuristics at both the sender and receiver to reduce the impact of this.

As described in Section 3.9, a Maximum Packet Size (MPS) is maintained for an MP-DCCP connection. If MP-DCCP exposes a minimum MPS across all paths, any change to one path impacts the sender for all paths. To mitigate attacks that seek to force a low MPS, MP-DCCP could detect an attempt to reduce the MPS less than a minimum MPS, and then stop using these paths.

## 5. Interactions with Middleboxes

Issues from interaction with on-path middleboxes such as NATs, firewalls, proxies, intrusion detection systems (IDSs), and others have to be considered for all extensions to standard protocols since otherwise unexpected reactions of middleboxes may hinder its deployment. DCCP already provides means to mitigate the potential impact of middleboxes, also in comparison to TCP (see [RFC4043], Section 16). When both hosts are located behind a NAT or firewall entity, specific measures have to be applied such as the [RFC5596] specified simultaneous-open technique that update the (traditionally asymmetric) connection-establishment procedures for DCCP. Further standardized technologies addressing middleboxes operating as NATs are provided in [RFC5597].

[RFC6773] specifies UDP Encapsulation for NAT Traversal of DCCP sessions, similar to other UDP encapsulations such as for SCTP [RFC6951]. Future specifications by the IETF could specify other methods for DCCP encapsulation.

The security impact of MP-DCCP aware middleboxes is discussed in Section 4.

## 6. Implementation

The approach described above has been implemented in open source across different testbeds and a new scheduling algorithm has been extensively tested. Also, demonstrations of a laboratory setup have been executed and have been published at [multipath-dccp.org].

## 7. Acknowledgments

[RFC8684] defines Multipath TCP and provided important inputs for this specification.

The authors gratefully acknowledge significant input into this document from Dirk von Hugo, Nathalie Romo Moreno, Omar Nassef, Mohamed Boucadair, Simone Ferlin, Olivier Bonaventure, Gorry Fairhurst and Behcet Sarikaya.

## 8. IANA Considerations

This section provides guidance to the Internet Assigned Numbers Authority (IANA) regarding registration of values related to the MP extension of the DCCP protocol in accordance with the RFC Required policy of [RFC8126], Section 4.7. This document defines one new value which is requested to be allocated in the IANA DCCP Feature Numbers registry and three new registries to be allocated in the DCCP registry group.

### 8.1. New Multipath Capable DCCP feature

This document requests IANA to assign a new DCCP feature parameter for negotiating the support of multipath capability for DCCP sessions between hosts as described in Section 3. The following entry in Table 6 should be added to the Feature Numbers registry in the DCCP registry group according to [RFC4340], Section 19.4. under the "DCCP Protocol" heading.

| Value | Feature Name | Specification |
|---|---|---|
| 10 suggested | Multipath Capable | [ThisDocument] |

Table 6: Addition to DCCP Feature Numbers registry

Note to RFC Editor: Please replace [ThisDocument] with a reference to the final RFC

### 8.2. New MP-DCCP version registry

Section 3.1 specifies the new 1-byte entry above includes a 4-bit part to specify the version of the used MP-DCCP implementation. This document requests IANA to create a new ’MP-DCCP Versions’ registry within the DCCP registry group to track the MP-DCCP version. The initial content of this registry is as follows:

| Version | Value | Specification |
|---|---|---|
| 0 | 0000 suggested | [ThisDocument] |
| 1-15 | unassigned | |

Table 7: MP-DCCP Versions Registry

Note to RFC Editor: Please replace [ThisDocument] with a reference to the final RFC

Future MP-DCCP versions 1 to 15 are assigned from this registry using the RFC Required policy (Section 4.7 of [RFC8126]).

### 8.3. New Multipath option and registry

This document requests IANA to assign value 46 in the DCCP "Option Types" registry to "Multipath Options", as described in Section 3.2.

IANA is requested to create a new ’Multipath Options’ registry within the DCCP registry group. The following entries in Table 8 should be added to the new ’Multipath Options’ registry. The registry in Table 8 has an upper boundary of 255 in the numeric value field.

| Multipath Option | Name | Description | Reference |
|---|---|---|---|
| MP_OPT=0 | MP_CONFIRM | Confirm reception/ processing of an MP_OPT option | Section 3.2.1 |
| MP_OPT=1 | MP_JOIN | Join subflow to an existing MP-DCCP connection | Section 3.2.2 |
| MP_OPT=2 | MP_FAST_CLOSE | Close an MP-DCCP connection unconditionally | Section 3.2.3 |
| MP_OPT=3 | MP_KEY | Exchange key material for MP_HMAC | Section 3.2.4 |
| MP_OPT=4 | MP_SEQ | Multipath sequence number | Section 3.2.5 |
| MP_OPT=5 | MP_HMAC | Hash-based message auth. code for MP-DCCP | Section 3.2.6 |
| MP_OPT=6 | MP_RTT | Transmit RTT values and calculation parameters | Section 3.2.7 |
| MP_OPT=7 | MP_ADDADDR | Advertise | Section |

| | | | | |
|---|---|---|---|---|
| | | additional address(es)/port(s) | 3.2.8 | |
| MP_OPT=8 | MP_REMOVEADDR | Remove address(es)/ port(s) | Section 3.2.9 | |
| MP_OPT=9 | MP_PRIO | Change subflow priority | Section 3.2.10 | |
| MP_OPT=10 | MP_CLOSE | Close an MP-DCCP connection | Section 3.2.11 | |
| MP_OPT=11 | MP_EXP | Experimental option for private use | Section 3.2.12 | |
| MP_OPT>11 | Unassigned | Reserved for future Multipath options | | |

Table 8: Multipath Options registry

Future Multipath options with MP_OPT>11 are assigned from this registry using the RFC Required policy (Section 4.7 of [RFC8126]).

## 8.4. New DCCP Reset Code

IANA is requested to assign a new DCCP-Reset Code value 13 suggested in the DCCP-Reset Codes Registry, with the short description "Abrupt MP termination". Use of this reset code is defined in section Section 3.2.3.

## 8.5. New Multipath Key Type registry

IANA is requested to assign for this version of the MP-DCCP protocol a new ’Multipath Key Type’ registry containing two different suboptions to the MP_KEY option to identify the MP_KEY Key types in terms of 8-bit values as specified in Section 3.2.4 according to the entries in Table 9 below. Values in range 3-254 (decimal) inclusive remain unassigned in this here specified version 0 of the protocol and are assigned via RFC Required [RFC8126] in potential future versions of the MP-DCCP protocol.

| Type | Name | Meaning | Reference |
|---|---|---|---|
| 0 | Plain Text | Plain text key | Section 3.2.4 |
| 1-254 | Unassigned | Reserved for future use | Section 3.2.4 |
| 255 | Experimental | For private use only | Section 3.2.4 |

Table 9: Multipath Key Type registry with the MP_KEY Key Types for key data exchange on different paths

# 9. References

## 9.1. Normative References

[DCCP.Parameter] "IANA Datagram Congestion Control Protocol (DCCP) Parameters", n.d., <https://www.iana.org/assignments/dccp-parameters/dccp-parameters.xhtml>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/rfc/rfc4086>.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, DOI 10.17487/RFC4340, March 2006, <https://www.rfc-editor.org/rfc/rfc4340>.

[RFC6234] Eastlake 3rd, D. and T. Hansen, "US Secure Hash Algorithms (SHA and SHA-based HMAC and HKDF)", RFC 6234, DOI 10.17487/RFC6234, May 2011, <https://www.rfc-editor.org/rfc/rfc6234>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/rfc/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

## 9.2. Informative References

[I-D.amend-iccrg-multipath-reordering]
Amend, M. and D. Von Hugo, "Multipath sequence maintenance", Work in Progress, Internet-Draft, draft-amend-iccrg-multipath-reordering-03, 25 October 2021, <https://datatracker.ietf.org/doc/html/draft-amend-iccrg-multipath-reordering-03>.

[I-D.amend-tsvwg-dccp-udp-header-conversion]
Amend, M., Brunstrom, A., Kassler, A., and V. Rakocevic, "Lossless and overhead free DCCP - UDP header conversion (U-DCCP)", Work in Progress, Internet-Draft, draft-amend-tsvwg-dccp-udp-header-conversion-01, 8 July 2019, <https://datatracker.ietf.org/doc/html/draft-amend-tsvwg-dccp-udp-header-conversion-01>.

[IETF105.Slides]
Amend, M., "MP-DCCP for enabling transfer of UDP/IP traffic over multiple data paths in multi-connectivity networks", IETF105 , n.d., <https://datatracker.ietf.org/meeting/105/materials/slides-105-tsvwg-sessa-62-dccp-extensions-for-multipath-operation-00>.

[MP-DCCP.Paper]
Amend, M., Bogenfeld, E., Cvjetkovic, M., Rakocevic, V., Pieska, M., Kassler, A., and A. Brunstrom, "A Framework for Multiaccess Support for Unreliable Internet Traffic using Multipath DCCP", DOI 10.1109/LCN44214.2019.8990746, October 2019, <https://doi.org/10.1109/LCN44214.2019.8990746>.

[multipath-dccp.org]
"Multipath extension for DCCP", n.d., <https://multipath-dccp.org/>.

[OLIA] Khalili, R., Gast, N., Popovic, M., Upadhyay, U., and J. Le Boudec, "MPTCP is not pareto-optimal: performance issues and a possible solution", Proceedings of the 8th international conference on Emerging networking experiments and technologies, ACM , 2012.

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <https://www.rfc-editor.org/rfc/rfc2104>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <https://www.rfc-editor.org/rfc/rfc3711>.

[RFC4043] Pinkas, D. and T. Gindin, "Internet X.509 Public Key Infrastructure Permanent Identifier", RFC 4043, DOI 10.17487/RFC4043, May 2005, <https://www.rfc-editor.org/rfc/rfc4043>.

[RFC5238] Phelan, T., "Datagram Transport Layer Security (DTLS) over the Datagram Congestion Control Protocol (DCCP)", RFC 5238, DOI 10.17487/RFC5238, May 2008, <https://www.rfc-editor.org/rfc/rfc5238>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/rfc/rfc5280>.

[RFC5596] Fairhurst, G., "Datagram Congestion Control Protocol (DCCP) Simultaneous-Open Technique to Facilitate NAT/ Middlebox Traversal", RFC 5596, DOI 10.17487/RFC5596, September 2009, <https://www.rfc-editor.org/rfc/rfc5596>.

[RFC5597] Denis-Courmont, R., "Network Address Translation (NAT) Behavioral Requirements for the Datagram Congestion Control Protocol", BCP 150, RFC 5597, DOI 10.17487/RFC5597, September 2009, <https://www.rfc-editor.org/rfc/rfc5597>.

[RFC6356] Raiciu, C., Handley, M., and D. Wischik, "Coupled Congestion Control for Multipath Transport Protocols", RFC 6356, DOI 10.17487/RFC6356, October 2011, <https://www.rfc-editor.org/rfc/rfc6356>.

[RFC6773] Phelan, T., Fairhurst, G., and C. Perkins, "DCCP-UDP: A Datagram Congestion Control Protocol UDP Encapsulation for NAT Traversal", RFC 6773, DOI 10.17487/RFC6773, November 2012, <https://www.rfc-editor.org/rfc/rfc6773>.

[RFC6904] Lennox, J., "Encryption of Header Extensions in the Secure Real-time Transport Protocol (SRTP)", RFC 6904, DOI 10.17487/RFC6904, April 2013, <https://www.rfc-editor.org/rfc/rfc6904>.

[RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, DOI 10.17487/RFC6951, May 2013, <https://www.rfc-editor.org/rfc/rfc6951>.

[RFC7323] Borman, D., Braden, B., Jacobson, V., and R. Scheffenegger, Ed., "TCP Extensions for High Performance", RFC 7323, DOI 10.17487/RFC7323, September 2014, <https://www.rfc-editor.org/rfc/rfc7323>.

[RFC8041] Bonaventure, O., Paasch, C., and G. Detal, "Use Cases and Operational Experience with Multipath TCP", RFC 8041, DOI 10.17487/RFC8041, January 2017, <https://www.rfc-editor.org/rfc/rfc8041>.

[RFC8684] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 8684, DOI 10.17487/RFC8684, March 2020, <https://www.rfc-editor.org/rfc/rfc8684>.

[RFC9293] Eddy, W., Ed., "Transmission Control Protocol (TCP)", STD 7, RFC 9293, DOI 10.17487/RFC9293, August 2022, <https://www.rfc-editor.org/rfc/rfc9293>.

[TS23.501] 3GPP, "System architecture for the 5G System; Stage 2; Release 16", December 2020, <https://www.3gpp.org/ftp//Specs/archive/23_series/23.501/23501-g70.zip>.

## Appendix A. Differences from Multipath TCP

This appendix is Informative.

Multipath DCCP is similar to Multipath TCP [RFC8684], in that it extends the related basic DCCP transport protocol [RFC4340] with multipath capabilities in the same way as Multipath TCP extends TCP [RFC9293]. However, because of the differences between the underlying TCP and DCCP protocols, the transport characteristics of MPTCP and MP-DCCP are different.

Table 10 compares the protocol characteristics of TCP and DCCP, which are by nature inherited by their respective multipath extensions. A major difference lies in the delivery of payload, which is for TCP an exact copy of the generated byte-stream. DCCP behaves differently and does not guarantee to deliver any payload nor the order of delivery. Since this is mainly affecting the receiving endpoint of a TCP or DCCP communication, many similarities on the sender side can be identified. Both transport protocols share the 3-way initiation of a communication and both employ congestion control to adapt the sending rate to the path characteristics.

| Feature | TCP | DCCP |
|---|---|---|
| Full-Duplex | yes | yes |
| Connection-Oriented | yes | yes |
| Header option space | 40 bytes | < 1008 bytes or PMTU |
| Data transfer | reliable | unreliable |
| Packet-loss handling | re-transmission | report only |
| Ordered data delivery | yes | no |
| Sequence numbers | one per byte | one per PDU |
| Flow control | yes | no |
| Congestion control | yes | yes |
| ECN support | yes | yes |
| Selective ACK | yes | depends on congestion control |
| Fix message boundaries | no | yes |
| Path MTU discovery | yes | yes |
| Fragmentation | yes | no |
| SYN flood protection | yes | no |
| Half-open connections | yes | no |

Table 10: TCP and DCCP protocol comparison

Consequently, the multipath features, shown in Table 11, are the same, supporting volatile paths having varying capacity and latency, session handover and path aggregation capabilities. All of them profit by the existence of congestion control.

| Feature | MPTCP | MP-DCCP |
|---|---|---|
| Volatile paths | yes | yes |
| Session handover | yes | yes |
| Path aggregation | yes | yes |
| Data reordering | yes | optional |
| Expandability | limited by TCP header | flexible |

Table 11: MPTCP and MP-DCCP protocol comparison

Therefore, the sender logic is not much different between MP-DCCP and MPTCP.

The receiver side for MP-DCCP has to deal with the unreliable delivery provided by DCCP. The multipath sequence numbers included in MP-DCCP (see Section 3.2.5) facilitates adding optional mechanisms for data stream packet reordering at the receiver. Information from the MP_RTT multipath option (Section 3.2.7), DCCP path sequencing and the DCCP Timestamp Option provide further means for advanced reordering approaches, e.g., as proposed in [I-D.amend-iccrg-multipath-reordering]. Such mechanisms do, however, not affect interoperability and are not part of the MP-DCCP protocol. Many applications that use unreliable transport protocols can also inherently process out-of-sequence data (e.g., through adaptive audio and video buffers), and so additional reordering support might not be necessary. The addition of optional reordering mechanisms are likely to be needed when the different DCCP subflows are routed across paths with different latencies. In theory, applications using DCCP are aware that packet reordering could occur, because DCCP does not provide mechanisms to restore the original packet order.

In contrast to TCP, the receiver processing for MPTCP adopted a rigid "just wait" approach, because TCP guarantees reliable in-order delivery.

## Authors' Addresses

Markus Amend (editor)
Deutsche Telekom
Deutsche-Telekom-Allee 9
64295 Darmstadt
Germany

Email: Markus.Amend@telekom.de

Anna Brunstrom
Karlstad University
Universitetsgatan 2
SE-651 88 Karlstad
Sweden
Email: anna.brunstrom@kau.se

Andreas Kassler
Karlstad University
Universitetsgatan 2
SE-651 88 Karlstad
Sweden
Email: andreas.kassler@kau.se

Veselin Rakocevic
City, University of London
Northampton Square
London
United Kingdom
Email: veselin.rakocevic.1@city.ac.uk

Stephen Johnson
BT
Adastral Park
Martlesham Heath
IP5 3RE
United Kingdom
Email: stephen.h.johnson@bt.com