Workgroup: Transport Area Working Group
Internet-Draft:
draft-ietf-tsvwg-multipath-dccp-05
Published: 8 July 2022
Intended Status: Experimental
Expires: 9 January 2023
Authors: M. Amend, Ed.
DT
A. Kassler
Karlstad University
S. Johnson
BT
A. Brunstrom
Karlstad University
V. Rakocevic
City University of London

DCCP Extensions for Multipath Operation with Multiple Addresses

Abstract

DCCP communication as defined in [RFC4340] is restricted to a single path per connection, yet multiple paths often exist between peers. The simultaneous use of these multiple paths for a DCCP session could improve resource usage within the network and, thus, improve user experience through higher throughput and improved resilience to network failures. Use cases for a Multipath DCCP (MP-DCCP) are mobile devices (handsets, vehicles) and residential home gateways simultaneously connected to distinct paths as, e.g., a cellular link and a WiFi link or to a mobile radio station and a fixed access network. Compared to existing multipath protocols such as MPTCP, MP-DCCP provides specific support for non-TCP user traffic as UDP or plain IP. More details on potential use cases are provided in [website], [slide], and [paper]. All these use cases profit from an Open Source Linux reference implementation provided under [website].

This document presents a set of extensions to traditional DCCP to support multipath operation. Multipath DCCP provides the ability to simultaneously use multiple paths between peers. The protocol offers the same type of service to applications as DCCP and it provides the components necessary to establish and use multiple DCCP subflows across potentially disjoint paths.

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 9 January 2023.

Copyright Notice

Table of Contents

1. Introduction

Multipath DCCP (MP-DCCP) is a set of extensions to regular DCCP [RFC4340], i.e., the Datagram Congestion Control Protocol denoting a transport protocol that provides bidirectional unicast connections of congestion-controlled unreliable datagrams. A multipath extension to DCCP enables the transport of user data across multiple paths simultaneously. This is beneficial to applications that transfer fairly large amounts of data, due to the possibility to aggregate capacity of the multiple paths. In addition, it enables to tradeoff timeliness and reliability, which is important for low latency applications that do not require guaranteed delivery services such as Audio/Video streaming. DCCP multipath operation has been suggested in the context of 3GPP work on 5G multi-access solutions [I-D.amend-tsvwg-multipath-framework-mpdccp] and for hybrid access networks [I-D.lhwxz-hybrid-access-network-architecture][I-D.muley-network-based-bonding-hybrid-access]. It can be applied for load-balancing, seamless session handover, and aggregation purposes (referred to as ATSSS; Access steering, switching, and splitting in 3GPP terminology [TS23.501]).

This document presents the protocol changes required to add multipath capability to DCCP; specifically, those for signaling and setting up multiple paths ("subflows"), managing these subflows, reordering of data, and termination of sessions. DCCP, as stated in [RFC4340] does not provide reliable and ordered delivery. Consequently, multiple application subflows may be multiplexed over a single DCCP connection with no inherent performance penalty for application subflows that do not require in-ordered delivery. DCCP does not provide built-in support for those multiple application subflows.

In the following, use of the term subflow will refer to physical separate DCCP subflows transmitted via different paths, but not to application subflows. Application subflows are differing content-wise by source and destination port per application as, for example, enabled by Service Codes introduced to DCCP in [RFC5595], and those application subflows can be multiplexed over a single DCCP connection. For sake of consistency we assume that only a single application is served by a DCCP connection here as shown in Figure 1

while use of that feature should not impact DCCP operation on each single path as noted in ([RFC5595], sect. 2.4).

1.1. Multipath DCCP in the Networking Stack

MP-DCCP operates at the transport layer and aims to be transparent to both higher and lower layers. It is a set of additional features on top of standard 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.

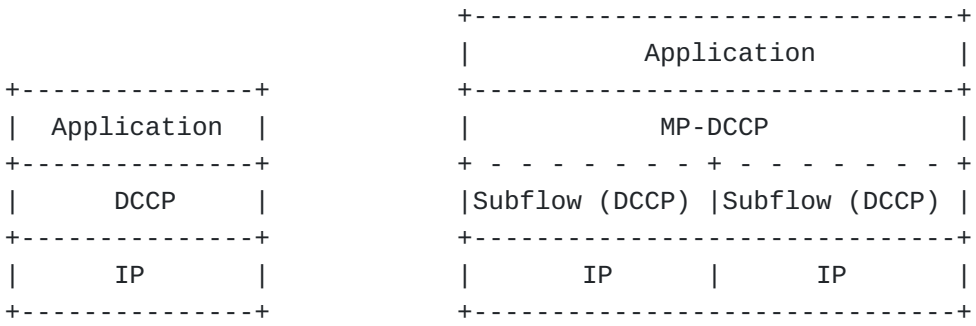


Figure 1: Comparison of Standard DCCP and MP-DCCP Protocol Stacks

1.2. Terminology

Throughout this document we make use of terms that are either specific for multipath transport or are defined in the context of MP-DCCP, similar to [RFC8684], 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/ port pairs.

Subflow: A flow of DCCP segments operating over an individual path, which forms part of a larger MP-DCCP connection. A subflow is started and terminated similar to a regular (single-path) DCCP connection. The term subflow can also be used to refer to an MP-DCCP connection with a single path.

(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.

Token: A locally unique identifier given to a multipath connection by a host. May also be referred to as a "Connection ID".

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

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

1.3. MP-DCCP Concept

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

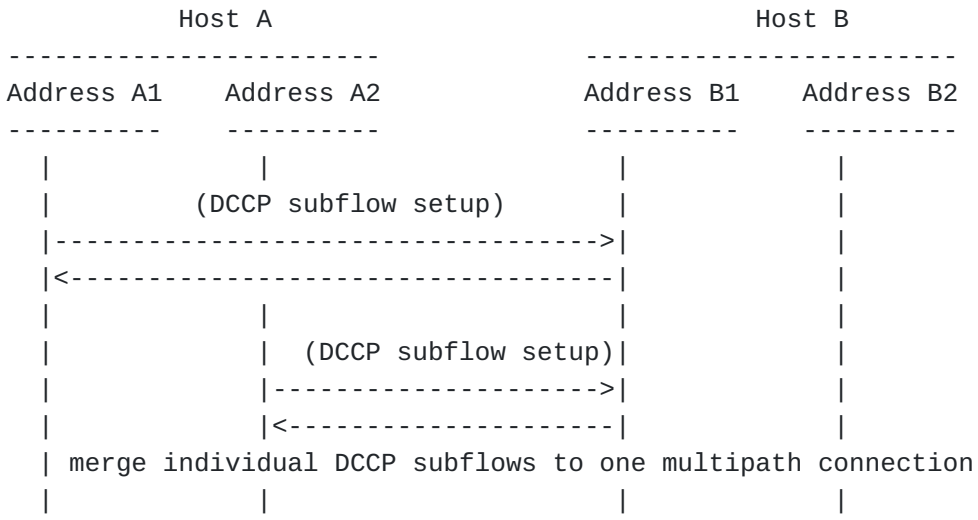


Figure 2: Example MP-DCCP Usage Scenario

*An MP-DCCP connection begins with a 4-way handshaking procedure, between 2 hosts as described in Section 3.3. In Figure 2 an MP-DCCP connection is established between addresses A1 and B1 on Hosts A and B, respectively.

*If extra paths are available, additional DCCP subflows are created on these paths and are attached to the existing MP-DCCP session, which continues to appear as a single connection to the applications at both ends. The creation of an additional DCCP subflow is illustrated between Address A2 on Host A and Address B1 on Host B.

*MP-DCCP identifies multiple paths by the presence of multiple addresses at hosts. Combinations of these multiple addresses equate to the additional paths. In the example, other potential paths that could be set up are A1<->B2 and A2<->B2. Although this additional session is shown as being initiated from A2, it could equally have been initiated from B1 or B2.

*The discovery and setup of additional subflows will be achieved through a path management method; this document describes supportive measures by which a host can initiate new subflows and available addresses can be signaled between peers. The definition

of a path management method is however out of scope of this document.

*MP-DCCP adds connection-level sequence numbers and exchange of RTT information to enable optional reordering functionalities.

*Subflows are terminated as regular DCCP connections, as described in ([RFC4340] section 8.3). The MP-DCCP connection is terminated by a connection-level DCCP-CloseReq or DCCP-Close message.

1.4. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Operation Overview

DCCP according to RFC 4340 [RFC4340] allows multiple application subflows to be multiplexed over a single DCCP connection with potentially same performance. However, DCCP does not provide built-in support for multiple subflows and the Congestion Manager (CM) [RFC3124], as a generic multiplexing facility, can not fully support multiple congestion control mechanisms for multiple DCCP flows between same source and destination addresses.

The proposed extension of DCCP towards Multipath-DCCP (MP-DCCP) is described in detail in Section 3.

As a high level overview on the key functionality of MP-DCCP, the data stream from a DCCP application is split by MP-DCCP operation into one or more subflows which can be transmitted via different also physically isolated paths. The corresponding control information allows the receiver to optionally re-assemble and deliver the received data in the right order to the recipient application. 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 the MP-DCCP protocol. The following sections define MP-DCCP behavior in detail.

The Multipath Capability for MP-DCCP can be negotiated with a new DCCP feature, as described and fully specified in Section 3. Once negotiated, all subsequent MP-DCCP operations are signalled with a variable length multipath-related option, as described in Section 3.1.

A Multipath DCCP connection provides a bidirectional byte-stream between two hosts exchanging data as in standard DCCP manner thus not requiring any change to the applications. However, Multipath DCCP enables the hosts to use different paths with different IP

addresses to transport the packets of the 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 exchange of performance parameters. The number of concurrent DCCP subflows can vary during the lifetime of the Multipath DCCP connection. All MP-DCCP operations are signaled with MP-DCCP options described in detail in {#MP_OPT}.

3. MP-DCCP Protocol

The DCCP protocol feature list ([RFC4340] section 6.4) will be enhanced by a new Multipath related feature with Feature number 10, as shown in Table 1.

Number	Meaning	Rule	Rec'n Value	Initial Req'd
0	Reserved			
1	Congestion Control ID (CCID)	SP	2	Y
2	Allow Short Seqnos	SP	0	Y
3	Sequence Window	NN	100	Y
4	ECN Incapable	SP	0	N
5	Ack Ratio	NN	2	N
6	Send Ack Vector	SP	0	N
7	Send NDP Count	SP	0	N
8	Minimum Checksum Coverage	SP	0	N
9	Check Data Checksum	SP	0	N
10	Multipath Capable	SP	0	N
11-127	Reserved			
128-255	CCID-specific features			

Table 1: Proposed Feature Set

Rec'n Rule: The reconciliation rule used for the feature. SP means server-priority, NN means non-negotiable.

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 (see Section 15), and it is safe to respond to Change options for the feature with empty Confirm options. Of course, a CCID might require the feature; a DCCP that implements CCID 2 MUST support Ack Ratio and Send Ack Vector, for example.

The DCCP protocol options as defined in ([RFC4340] section 5.8) and ([RFC5634] section 2.2.1) will be enhanced by a new Multipath

related variable-length option with option type 46, as shown in Table 2.

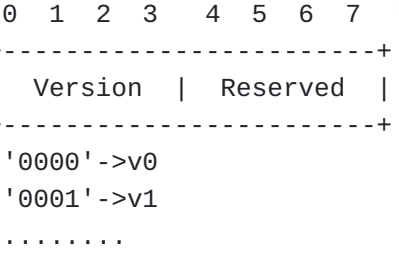
Type	Option Length	Meaning	DCCP-Data?
0	1	Padding	Y
1	1	Mandatory	N
2	1	Slow Receiver	Y
3-31	1	Reserved	
32	variable	Change L	N
33	variable	Confirm L	N
34	variable	Change R	N
35	variable	Confirm R	N
36	variable	Init Cookie	N
37	3-8	NDP Count	Y
38	variable	Ack Vector [Nonce 0]	N
39	variable	Ack Vector [Nonce 1]	N
40	variable	Data Dropped	N
41	6	Timestamp	Y
42	6/8/10	Timestamp Echo	Y
43	4/6	Elapsed Time	N
44	6	Data Checksum	Y
45	8	Quick-Start Response	?
46	variable	Multipath	Y
47-127	variable	Reserved	
128-255	variable	CCID-specific options	-

Table 2: Proposed Option Set

3.1. Multipath Capable Feature

DCCP endpoints are multipath-disabled by default and multipath capability can be negotiated with the Multipath Capable Feature.

Multipath Capable has feature number 10 and has length of one-byte. The leftmost four bits are used to specify a compatible version of the MP-DCCP implementation (0 for this specification). The following four bits are reserved for further use.



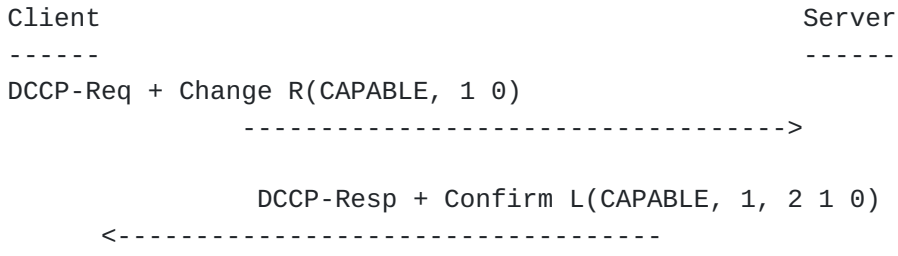
Multipath Capable follows the server-priority reconciliation rule described in ([RFC4340] section 6.3.1), which allows multiple versions to be specified in order of priority.

The negotiation MUST be done as part of the initial handshake procedure as described in Section 3.3, and no subsequent re-negotiation of the Multipath Capable feature is allowed on the same MP connection.

Client MUST include a Change R option during the initial handshake request to supply a list of supported protocol versions, ordered by preference.

Server MUST include a Confirm L option in the subsequent response to agree on a version to be used chosen from the Client list, followed by its own supported version(s) ordered by preference.

For example:



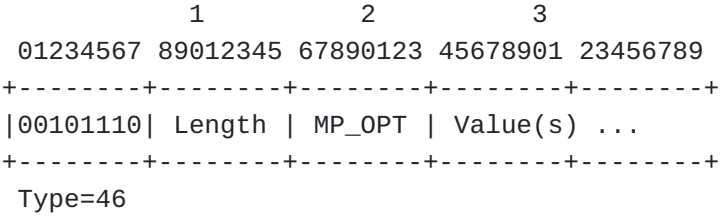
1. Client indicates support for both versions 1 and 0, with preference for version 1

2. Server agrees on using version 1, and supply its own preference list.

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

Any subflow addition to an existing MP connection MUST use the same version negotiated for the first subflow.

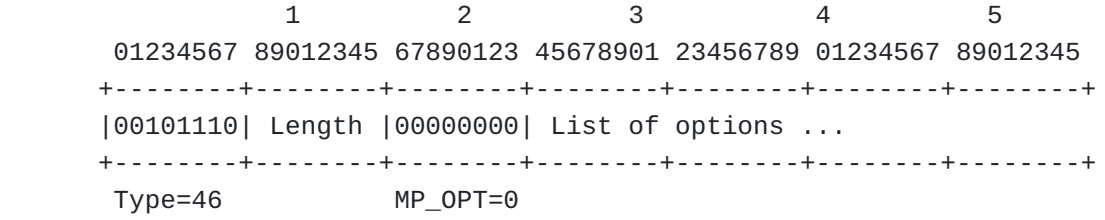
3.2. Multipath Option



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 path to an existing MP-DCCP connection
46	3	2 =MP_FAST_CLOSE	Close MP-DCCP connection unconditionally
46	var	3 =MP_KEY	Exchange key material for MP_HMAC
46	7	4 =MP_SEQ	Multipath Sequence Number
46	23	5 =MP_HMAC	HMA Code for authentication
46	12	6 =MP_RTT	Transmit RTT values
46	var	7 =MP_ADDADDR	Advertise additional Address
46	4	8 =MP_REMOVEADDR	Remove Address
46	4	9 =MP_PRIO	Change Subflow Priority
46	3	10 =MP_CLOSE	Close MP-DCCP subflow

Table 3: MP_OPT Option Types

3.2.1. MP_CONFIRM

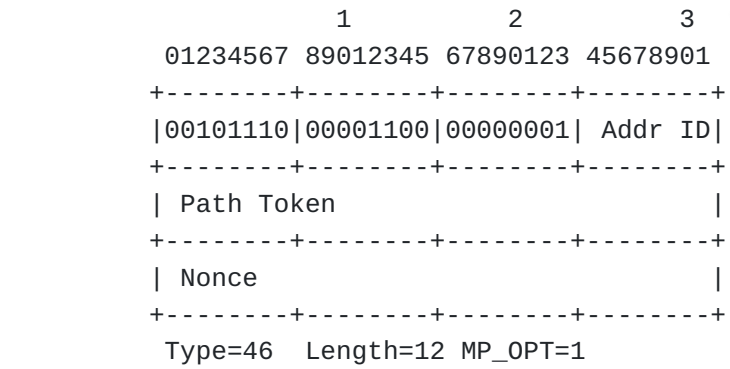


MP_CONFIRM is used to send confirmation of reception and processing of the multipath options that require it (see Table 4). Such options will be retransmitted by the sender until this receives the corresponding MP_CONFIRM. The length and sending path of the MP_CONFIRM are dependent on the confirmed options, which will be copied verbatim and appended as list of options.

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

Table 4: Multipath options requiring confirmation

3.2.2. MP_JOIN



The MP_JOIN option is used to add a new subflow to an existing MP-DCCP connection. The Path Token is the SHA256 hash of the derived key (d-key), which was previously exchanged with the MP_KEY option. MP_HMAC MUST be set when using MP_JOIN to provide authentication (See MP_HMAC for details). Also MP_KEY MUST be set to provide key material for authentication purposes.

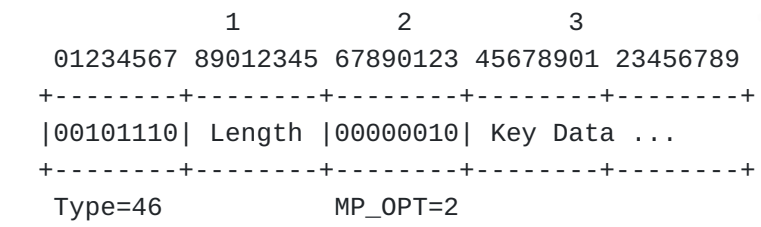
The MP_JOIN option includes an "Address ID". This is an identifier generated by the sender of the option, used to identify the source address of this packet, even if the IP header has been changed in transit by a middlebox. The numeric value of this field is generated by the sender and must map uniquely to a source IP address for the sending host. The Address ID allows address removal (Section 3.2.9) without needing to know what the source address at the receiver is, 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 has to 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 Token, the Nonce value builds the basis to calculate the HMAC used in the handshaking process as described in Section 3.3.

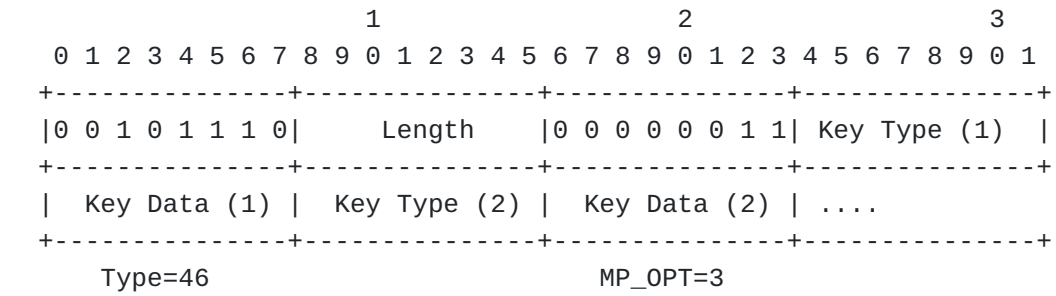
3.2.3. MP_FAST_CLOSE

Regular DCCP has the means of sending a Close or Reset signal to abruptly close a connection. With MP-DCCP, a regular Close or Reset only has the scope of the subflow; it will only close the applicable subflow and will not affect the remaining subflows concurrently in use on other paths. MP-DCCP's connection will stay alive at the data level, in order to permit break-before-make handover between subflows. It is therefore necessary to provide an MP-DCCP-level "Reset" to allow the abrupt closure of the whole MP-DCCP connection; this is done via the MP_FAST_CLOSE option.



For being effective, the MP_FAST_CLOSE must be sent from an initiating host on all subflows as part of a DCCP-Reset packet with Reset Code 13. To protect unauthorized shutdown of a multi-path connection, the selected Key Data of the peer host during the handshaking procedure is carried by the MP_FAST_CLOSE option. With completion of this step, the initiating host can tear down the subflows and the multi-path connection immediately terminates. Upon reception of the MP_FAST_CLOSE and validation of the Key Data at the peer host, a DCCP Reset packet is replied on all subflows to the initiating host again with Reset Code 13. The peer host can now close the whole MP-DCCP connection (it transitions directly to CLOSED state).

3.2.4. MP_KEY



The MP_KEY suboption is used to exchange key material between hosts. The Length varies between 12 and 68 Bytes for a single-key message, and up to 110 Bytes when all specified Key Types 0-2 are provided. The Key Type field is used to specify the type of the following key data. Key types are shown in Table 5.

Key Type	Key Length (Bytes)	Meaning
0 =Plain Text	8	Plain Text Key
1 =ECDHE-C25519-SHA256	32	ECDHE with SHA256 and Curve25519
2 =ECDHE-C25519-SHA512	64	ECDHE with SHA512 and Curve25519
3-255		Reserved

Table 5: MP_KEY Key Types

Plain Text

Key Material is exchanged in plain text between hosts, and the key parts (key-a, key-b) are used by each host to generate the derived key (d-key) by concatenating the two parts with the local key in front (e.g. hostA d-key(A)=(key-a+key-b), hostB d-key(B)=(key-b+key-a)).

ECDHE-SHA256-C25519

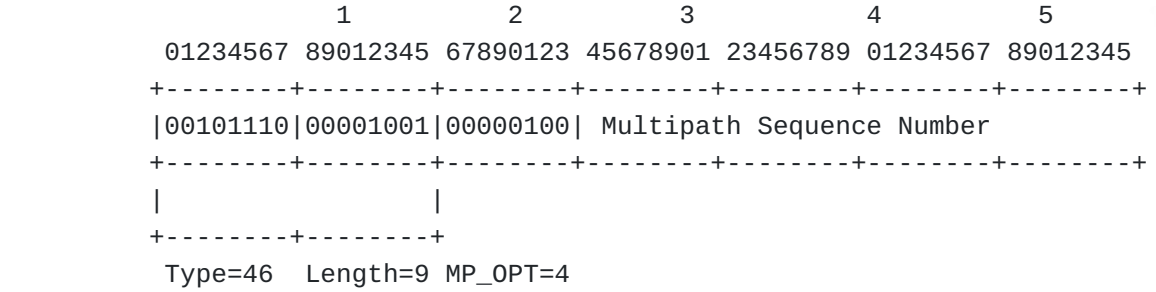
Public Key Material is exchanged via ECDHE key exchange with SHA256 and Curve 25519 to generate the derived key (d-key) from the shared secret.

ECDHE-SHA512-C25519

Public Key Material is exchanged via ECDHE key exchange with SHA512 and Curve 25519 to generate the derived key (d-key) from the shared secret.

Providing multiple keys is only permitted in the DCCP-Request message of the handshake procedure for the first subflow, and allows the hosts to agree on a single key type to be used as described in Section 3.3

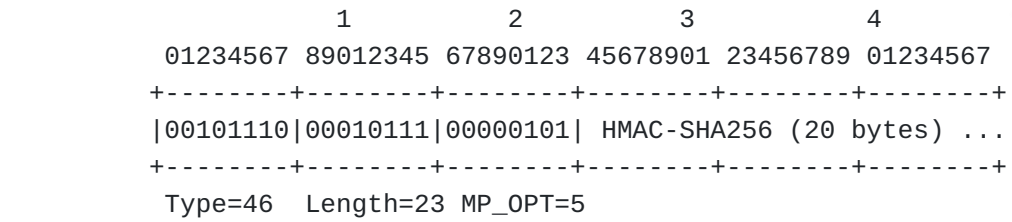
3.2.5. MP_SEQ



Type=46 Length=9 MP_OPT=4

The MP_SEQ option is used for end-to-end datagram-based sequence numbers of an MP-DCCP connection. The initial data sequence number (IDSN) SHOULD be set randomly. The MP_SEQ number space is different from the path individual sequence number space and MUST be sent with any DCCP-Data and DCCP-DataACK packet.

3.2.6. MP_HMAC

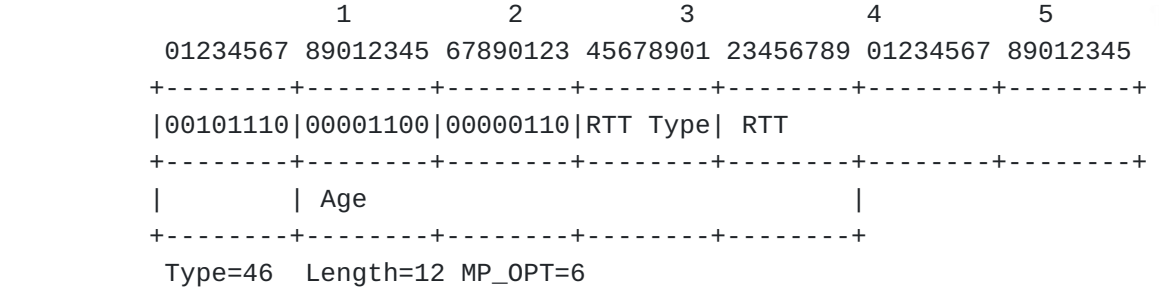


The MP_HMAC option is used to provide authentication for the MP_JOIN, MP_ADDADDR and MP_REMOVEADDR option. The HMAC code is generated according to [RFC2104] in combination with the SHA256 hash algorithm described in [RFC6234], with the output truncated to the leftmost 160 bits (20 bytes).

The "Key" used for the HMAC computation is the derived key (d-key) described in Section 3.2.4, while the HMAC "Message" is a concatenation of

- *MP_JOIN: The token and nonce of the MP_JOIN for which authentication shall be performed.
- *MP_ADDADDR: The Address ID with associated IP address and if defined port, otherwise two octets of value 0.
- *MP_REMOVEADDR: Solely the Address ID.

3.2.7. MP_RTT



The MP_RTT option is used to transmit RTT values in milliseconds and MUST belong to the path over which this information is transmitted. Additionally, the age of the measurement is specified in milliseconds.

The RTT and Age information is a 32-bit integer, which allows to cover a period of approximately 1193 hours.

Raw RTT (=0)
Raw RTT value of the last Datagram Round-Trip preferably provided by the CCID in use.

Min RTT (=1)
Min RTT value over a given period preferably provided by the CCID in use.

Max RTT (=2)
Max RTT value over a given period preferably provided by the CCID in use.

Smooth RTT (=3)
Averaged RTT value over a given period preferably provided by the CCID in use.

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.

3.2.8. MP_ADDADDR

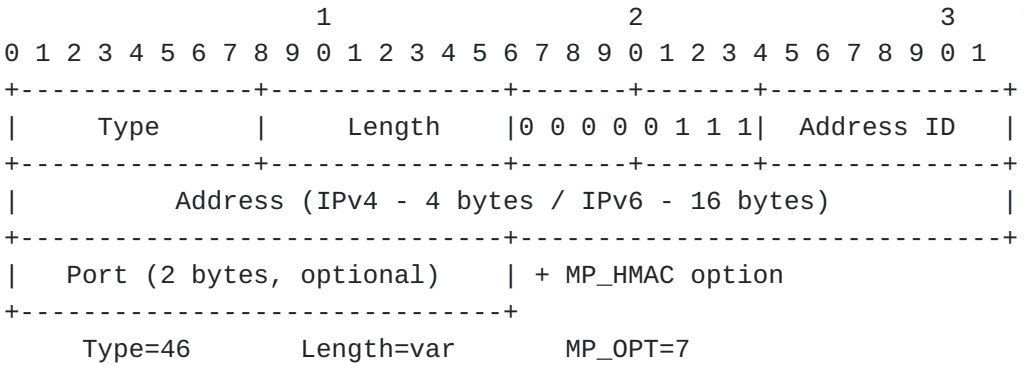
The MP_ADDADDR option announces additional addresses (and, optionally, ports) on which a host can be reached. This option can be used at any time during an existing DCCP connection, when the sender wishes to enable multiple paths and/or when additional paths become available. Multiple instances of this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

Length is variable depending on IPv4 or IPv6 and whether port number is used and is in range between 8 and 22 bytes.

The presence of the final 2 octets, specifying the DCCP port number to use, are optional and 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 may be cases (such as port-based load balancing) where the explicit specification of a different port is required. If no port is specified, MP-DCCP SHOULD attempt to connect to the specified address on the same port as is already in use by the subflow on which the MP_ADDADDR signal was sent.

Along with the MP_ADDADDR option 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 Key-A followed by Key-B, and in the case of Host B, Key-B followed by Key-A. These are the keys that were exchanged and selected in the original MP_KEY handshake. The message for the HMAC is the Address ID, IP address, and port that

precede the HMAC in the MP_ADDADDR option. If the port is not present in the MP_ADDADDR option, the HMAC message will nevertheless include 2 octets 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 SHOULD silently ignore the option.



Every address has an Address ID that can be used for uniquely identifying the address within a connection for address removal. 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 SHOULD be stored by the receiver in a data structure that gathers all the Address-ID-to-address mappings for a connection (identified by a token pair). In this way, there is a stored mapping between the Address ID, observed source address, and token pair for future processing of control information for a connection. Note that an implementation MAY discard incoming address advertisements at will, for example, for avoiding the required mapping state, or because advertised addresses are of no use to it (for example, IPv6 addresses when it has IPv4 only). Therefore, a host MUST treat address advertisements as soft state, and it MAY choose to refresh advertisements periodically.

Due to the proliferation of NATs, it is reasonably likely that one host may attempt to advertise private addresses. It is not desirable to prohibit this, since there may be cases where both hosts have additional interfaces on the same private network, and a host MAY want to advertise such addresses. 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 token that uniquely

identifies the connection to the receiving host. If the token is unknown, the host will return with a DCCP-Reset. In the unlikely event that the token is known, subflow setup will continue, but the HMAC exchange must occur for authentication. This will fail, and will provide sufficient protection against two unconnected hosts accidentally setting up a new subflow upon the signal of a private address. 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.

The reception of an MP_ADDADDR message is acknowledged using MP_CONFIRM (Section 3.2.1). Using this mechanism reliable exchange of address information is ensured.

A host can send an MP_ADDADDR message with an already assigned Address ID, but the Address MUST be the same as previously assigned to this Address ID, and the Port MUST be different from one already in use for this Address ID. If these conditions are not met, the receiver SHOULD silently ignore the MP_ADDADDR. A host wishing to replace an existing Address ID MUST first remove the existing one (Section 3.2.9).

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

3.2.9. MP_REMOVEADDR

If, during the lifetime of an MP-DCCP connection, a previously announced address becomes invalid (e.g., if the interface disappears), the affected host SHOULD announce this so that the peer can remove subflows related to this address.

This is achieved through the Remove Address (MP_REMOVEADDR) option which will remove a previously added address with an Address ID from a connection and terminate any subflows currently using that address.

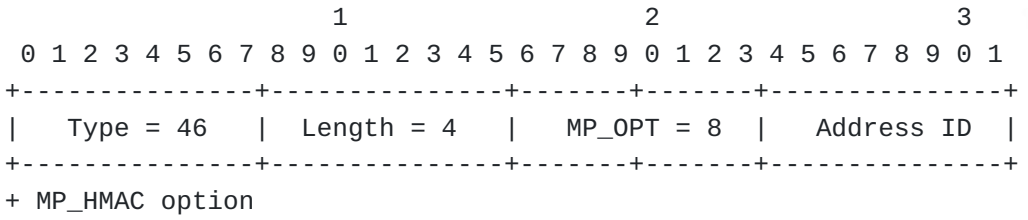
Along with the MP_REMOVEADDR option a MP_HMAC option MUST be sent for authentication. The truncated HMAC parameter present in this MP_HMAC option is the leftmost 20 bytes of an HMAC, negotiated and calculated as described in Section 3.2.6. In the same way as for MP_JOIN, the key for the HMAC algorithm, in the case of the message transmitted by Host A, will be Key-A followed by Key-B, and in the case of Host B, Key-B followed by Key-A. 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 the 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 removed 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 SHOULD silently ignore the option.

The reception of an MP_REMOVEADDR message is acknowledged using MP_CONFIRM (Section 3.2.1). Using this mechanism reliable exchange of address information is ensured.

The sending and receipt of this message SHOULD trigger the sending of DCCP-Close and DCCP-Reset by client and server, respectively on the affected subflow(s) (if possible), as a courtesy to cleaning up middlebox state, before cleaning up any local state.

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

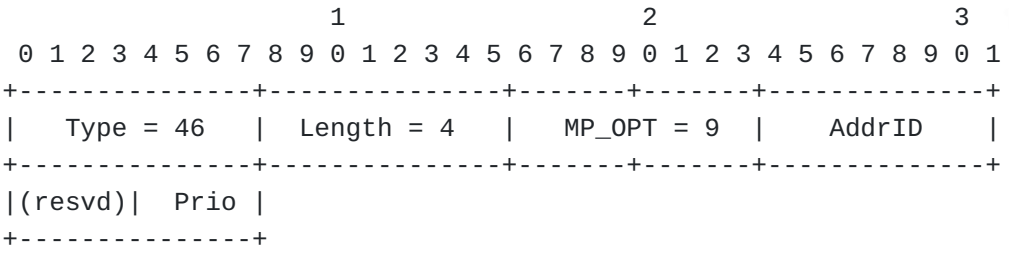


A subflow that is still functioning MUST be closed with a DCCP-Close or exchange as in regular DCCP, rather than using this option. For more information, see Section Section 3.4.

3.2.10. MP_PRIO

In the event that a single specific path out of the set of available paths shall be treated with higher priority compared to the others when making scheduling decisions for user plane traffic, a host may wish to signal such change in priority to the peer. One reason for such behavior is due to the different costs involved in using different paths (e.g., WiFi is free while cellular has limit on volume, 5G has higher energy consumption). Also, the priority of a path may be subject to dynamic changes, for example when the mobile runs out of battery, the usage of only a single path may be the preferred choice of the user. Therefore, the path priority should be considered as hints for the packet scheduler when making decisions which path to use for user plane traffic.

The MP_PRIO option, shown below, can be used to set a priority flag for the path which is specified by the AddrID field that uniquely identifies the path. The option can be sent over any path.



The following values are available for 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.

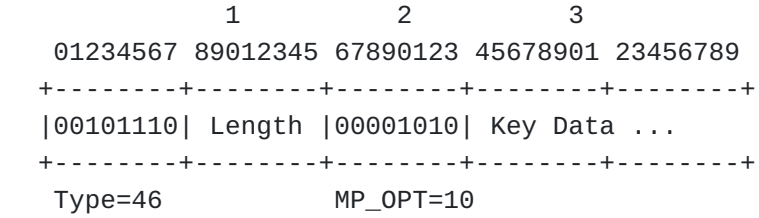
- *2: Secondary: do not use this path for traffic scheduling, if the other paths are good enough. The path will be used occasionally for increasing temporarily the available capacity, e.g. when primary paths are congested or are not available. This is the recommended setting for paths that have costs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

- *3 - 15: Primary: can use the path in any way deemed reasonable by peer. The path will always 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 more traffic 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 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 WiFi 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 only, if the Wi-Fi path is not available. 3) Setting Wi-Fi path to Primary and Cellular path to Primary. In this case, all packets can be scheduled over all paths at all time.

If not specified, the default behavior is, that a path can always be used for packet scheduling decisions (MP_PRIO=3), if the path has been established and added to an existing MP-DCCP connection. At least one path should have a MP_PRIO value greater or equal to one for it to be allowed to send on the connection. MP_PRIO is assumed to be exchanged reliably using the MP_CONFIRM mechanisms (see Table 4).

3.2.11. MP_CLOSE



For a graceful shutdown of a MP-DCCP connection, MP_CLOSE is used to communicate this to a 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). In the case where a DCCP-CloseReq is used, the following DCCP-Close MUST carry the MP_CLOSE as well. At the initiator of the DCCP-CloseReq all sockets, including the MP-DCCP connection socket, transition to CLOSEREQ state. To protect unauthorized shutdown of a multi-path connection, the selected Key Data of the peer host during the handshaking procedure MUST be carried by the MP_CLOSE option and validated by the peer host. Note, Key Data is different between MP_CLOSE option carried by DCCP-CloseReq or DCCP-Close.

On reception of a first DCCP-CloseReq carrying a 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. In this case, the MP-DCCP connection socket on the host sending the DCCP-Close reflects the state of the initial subflow used 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 a 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 a MP_FAST_CLOSE Section 3.2.3, no single-sided abrupt termination is applied.

3.3. MP-DCCP Handshaking Procedure

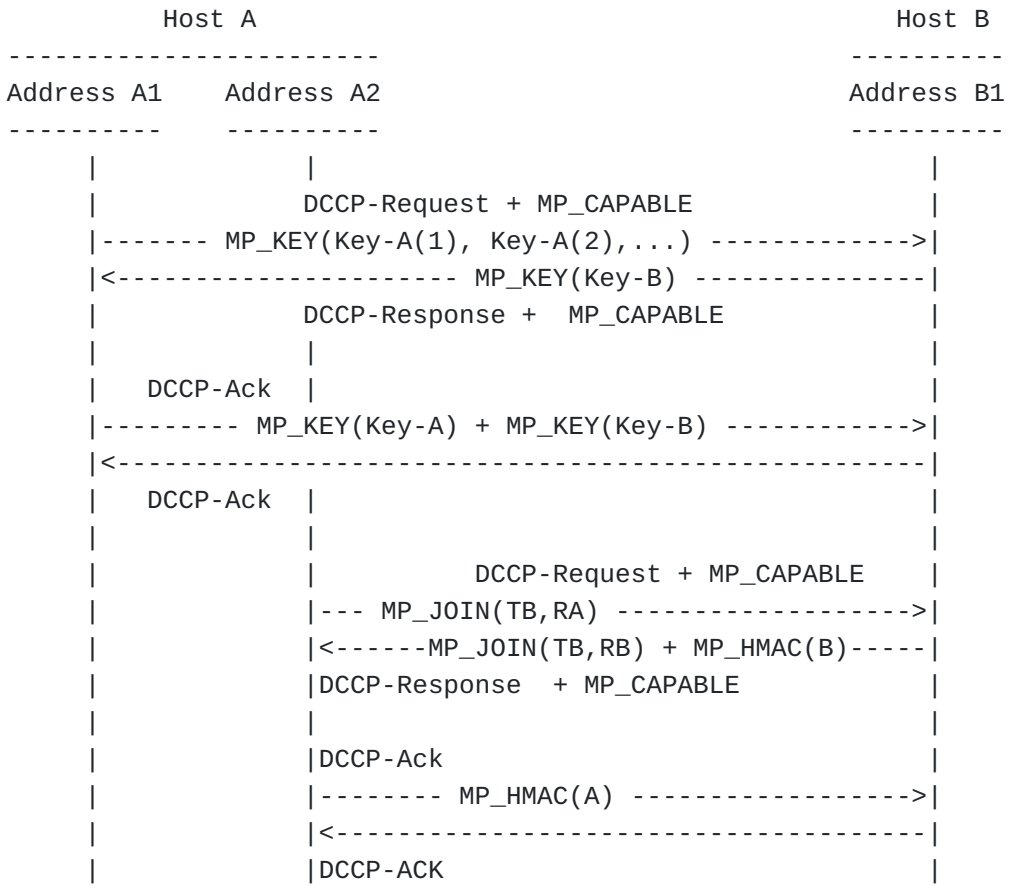


Figure 3: Example MP-DCCP Handshake

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 an Host-specific Key-A for each of supported types as described in Section 3.2.4

- Host B sends a DCCP-Response with Confirm feature for MP-Capable and the MP_Key option with a single Host-specific Key-B. The type of the key MUST be chosen from the list of supported types from the previous request

- Host A sends a DCCP-Ack with both Keys echoed to Host B.

- Host B sends a DCCP-Ack to confirm both keys and conclude the handshaking.

Host A MUST wait 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 Token TB, generated from the derived key by applying a SHA256 hash and truncating to the first 32 bits. 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 Token TB and a random nonce RB together with the computed MP_HMAC. The HMAC is calculated by taking the leftmost 20 bytes from the SHA256 hash of a HMAC code created by using token and nonce received with MP_JOIN(A) as message and the derived key described in Section 3.2.4 as key:

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

*Host A sends a DCCP-Ack with the HMAC computed for the DCCP-Response. The HMAC is calculated by taking the leftmost 20 bytes from the SHA256 hash of a HMAC code created by using token and 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-key(A), Msg=RA+RB)

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

3.4. Close a MP-DCCP connection

When a host wants to close an existing subflow but not the whole connection, it can initiate the regular DCCP connection termination procedure as described in [RFC4340], by using a DCCP-Close/DCCP-Reset on the subflow. This can optionally be preceded by a DCCP-CloseReq.

When a host wants to terminate the MP-DCCP connection, the recommended approach is to initiate the standard DCCP connection termination on each subflow with the first packet on each subflow carrying MP_CLOSE, as described in 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, a 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]
```

3.5. Fallback

When a subflow fails to operate within the MP-DCCP requirements, it is necessary to fall back to the safe operation. This may be either falling back to regular DCCP, 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 MP-DCCP options, faulty/non-supported MP-DCCP options or modification of payload data.

At the start of the 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 don't carry the MP_CAPABLE feature, the MP-DCCP connection will not be established and the handshake SHOULD fall back to regular DCCP or MUST be closed.

The same fallback SHOULD take place if the endpoints fail to agree on a protocol version to use during the Multipath Capable feature negotiation, which 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.

Similar procedure MUST be applied if the MP_KEY Section 3.2.4 Key Type cannot be negotiated, a final ACK carrying MP_KEY with wrong Key-A/Key-B is received or MP_KEY option is malformed.

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. Also non-verifiable MP_HMAC Section 3.2.6 or MP_JOIN Path Token Section 3.2.2 MUST lead to a subflow closing.

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 this happens in an MP-DCCP connection, the affected subflow can either be closed or other action can be taken.

3.6. Congestion Control Considerations

Senders MUST manage per-path congestion status, and SHOULD avoid to send more data on a given path than congestion control on that path allows.

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

3.7. Maximum Packet Size Considerations

A DCCP implementation MUST maintain 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. As per this definition a DCCP application interface SHOULD let the application discover the

current MPS, this is subject to ambiguity with potential different path MPS in a multi-path system. For compatibility reasons, a MP-DCCP implementation SHOULD always announce the minimum MPS across all paths.

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 or some other kind of 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.

As described in [RFC4340], DCCP provides protection against hijacking and limits the potential impact of some denial-of-service attacks, but DCCP provides no inherent protection against attackers' snooping on data packets. Regarding the security of MP-DCCP no additional risks should be introduced compared to regular DCCP. Thereof derived are the following key security requirements to be fulfilled by MP-DCCP:

- Provide a mechanism to confirm that parties involved in a subflow handshake are identical to those in the original connection setup.

- Provide verification that the new address to be included in a MP connection is valid for a peer to receive traffic at before using it.

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

In order to achieve these goals, MP-DCCP includes a hash-based handshake algorithm documented in Sections Section 3.2.4 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. To ease demultiplexing while not giving away any cryptographic material, future subflows use a truncated cryptographic hash of this key as the connection identification "token". The keys 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 are the same as in the original connection setup. It 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) at

both ends -- 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) will 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) will ensure that this does not succeed in setting 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.

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], sect. 16). In case, however, 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 NAT type middleboxes are covered by [RFC5597].

[RFC6773] specifies UDP Encapsulation for NAT Traversal of DCCP sessions, similar to other UDP encapsulations such as for SCTP [RFC6951]. The alternative U-DCCP approach proposed in [I-D.amend-tsvwg-dccp-udp-header-conversion] would reduce tunneling overhead. The handshaking procedure for DCCP-UDP header conversion or use of a DCCP-UDP negotiation procedure to signal support for DCCP-UDP header conversion would require encapsulation during the handshakes and use of two additional port numbers out of the UDP port number space, but would require zero overhead afterwards.

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 [website].

7. Acknowledgments

Due to the great spearheading work of the Multipath TCP authors in [RFC6824]/[RFC8684], some text passages for the -00 version of the draft were copied almost unmodified.

The authors gratefully acknowledge significant input into this document from Dirk von Hugo, Nathalie Romo Moreno and Omar Nassef.

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 [RFC8126]. This document defines one new value to DCCP feature list and one new DCCP Option with ten corresponding Subtypes as follows. This document defines 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" according to [RFC4340], Section 19.4. under the "DCCP Protocol" heading.

Value	Feature Name	Specification
0x10	MP-DCCP capability feature	Section 3.1

Table 6: Addition to DCCP Feature list Entries

This document defines a new DCCP protocol option of type=46 as described in Section 3.2 together with 10 additional sub-options. The following entries in Table 7 should be added to the "DCCP Protocol options" and assigned as "MP-DCCP sub-options", respectively.

Value	Symbol	Name	Reference
TBD or Type=46	MP_OPT	DCCP Multipath option	Section 3.2
TBD or MP_OPT=0	MP_CONFIRM	Confirm reception/ processing of an MP_OPT option	Section 3.2.1
TBD or MP_OPT=1	MP_JOIN	Join subflow to existing MP-DCCP connection	Section 3.2.2
TBD or MP_OPT=2	MP_FAST_CLOSE	Close MP-DCCP connection	Section 3.2.3
TBD or MP_OPT=3	MP_KEY	Exchange key material for MP_HMAC	Section 3.2.4

Value	Symbol	Name	Reference
TBD or MP_OPT=4	MP_SEQ	Multipath Sequence Number	Section 3.2.5
TBD or MP_OPT=5	MP_HMAC	Hash-based Message Auth. Code for MP-DCCP	Section 3.2.6
TBD or MP_OPT=6	MP_RTT	Transmit RTT values and calculation parameters	Section 3.2.7
TBD or MP_OPT=7	MP_ADDADDR	Advertise additional Address(es)/Port(s)	Section 3.2.8
TBD or MP_OPT=8	MP_REMOVEADDR	Remove Address(es)/ Port(s)	Section 3.2.9
TBD or MP_OPT=9	MP_PRIO	Change Subflow Priority	Section 3.2.10
TBD or MP_OPT=10	MP_CLOSE	Close MP-DCCP subflow	Section 3.2

Table 7: Addition to DCCP Protocol options and corresponding sub-options

In addition IANA is requested to assign a new DCCP Reset Code value 13 (or TBD) 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.

In addition IANA is requested to assign for this version of the MP-DCCP protocol three different sub options 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 8 below. Values in range 3-255 remain unspecified and are reserved for use in potential future versions of the MP-DCCP protocol.

Value	Key Type	Name or Meaning	Reference
TBD or 0	Plain Text	Plain Text Key	Section 3.2.4
TBD or 1	ECDHE-C25519-SHA256	ECDHE with SHA256 and Curve25519	Section 3.2.4
TBD or 2	ECDHE-C25519-SHA512	ECDHE with SHA512 and Curve25519	Section 3.2.4

Table 8: MP_KEY Key type sub-options for key data exchange on different paths

9. Informative References

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

[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://www.ietf.org/archive/id/draft-amend-tsvwg-dccp-udp-header-conversion-01.txt>.

[I-D.amend-tsvwg-multipath-framework-mpdccp]
Amend, M., Bogenfeld, E., Brunstrom, A., Kassler, A., and V. Rakocevic, "A multipath framework for UDP traffic over heterogeneous access networks", Work in Progress, Internet-Draft, draft-amend-tsvwg-multipath-framework-mpdccp-01, 8 July 2019, <https://www.ietf.org/archive/id/draft-amend-tsvwg-multipath-framework-mpdccp-01.txt>.

[I-D.lhwxz-hybrid-access-network-architecture]
Leymann, N., Heidemann, C., Wesserman, M., Xue, L., and M. Zhang, "Hybrid Access Network Architecture", Work in Progress, Internet-Draft, draft-lhwxz-hybrid-access-network-architecture-02, 13 January 2015, <https://www.ietf.org/archive/id/draft-lhwxz-hybrid-access-network-architecture-02.txt>.

[I-D.muley-network-based-bonding-hybrid-access]
Muley, P., Henderickx, W., Liang, G., Liu, H., Cardullo, L., Newton, J., Seo, S., Draznin, S., and B. Patil, "Network based Bonding solution for Hybrid Access", Work in Progress, Internet-Draft, draft-muley-network-based-bonding-hybrid-access-03, 22 October 2018, <https://www.ietf.org/archive/id/draft-muley-network-based-bonding-hybrid-access-03.txt>.

[OLIA] Khalili, R., Gast, N., Popovic, M., Upadhyay, U., and J.-Y. 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.

[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>.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <https://www.rfc-editor.org/info/rfc793>.

[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/info/rfc2104>.

[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/info/rfc2119>.

[RFC3124] Balakrishnan, H. and S. Seshan, "The Congestion Manager", RFC 3124, DOI 10.17487/RFC3124, June 2001, <https://www.rfc-editor.org/info/rfc3124>.

[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/info/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/info/rfc4043>.

[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/info/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/info/rfc4340>.

[RFC5595] Fairhurst, G., "The Datagram Congestion Control Protocol (DCCP) Service Codes", RFC 5595, DOI 10.17487/RFC5595, September 2009, <https://www.rfc-editor.org/info/rfc5595>.

[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/info/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/info/rfc5597>.

[RFC5634] Fairhurst, G. and A. Sathiaseelan, "Quick-Start for the Datagram Congestion Control Protocol (DCCP)", RFC 5634, DOI 10.17487/RFC5634, August 2009, <https://www.rfc-editor.org/info/rfc5634>.

[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/info/rfc6234>.

[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/info/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/info/rfc6773>.

[RFC6824] Ford, A., Raiciu, C., Handley, M., and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6824, DOI 10.17487/RFC6824, January 2013, <https://www.rfc-editor.org/info/rfc6824>.

[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/info/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/info/rfc6951>.

[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/info/rfc8126>.

[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/info/rfc8684>.

[slide] 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>.

[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>.

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

Appendix A. Differences from Multipath TCP

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 [RFC0793]. However, because of the differences between the underlying TCP and DCCP protocols, the transport characteristics of MPTCP and MP-DCCP are different.

Table 9 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 in a different way 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

Feature	TCP	DCCP
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 9: TCP and DCCP protocol comparison

Consequently, the multipath features, shown in Table 10, 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 10: 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 transport character of 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 described 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 deal with out-of-sequence data (e.g., through adaptive audio and video buffers), and so additional reordering support may not be necessary. The addition of optional reordering mechanisms are most 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 might happen, since DCCP has no mechanisms to prevent it.

The receiving process for MPTCP is on the other hand a rigid "just wait" approach, since TCP guarantees reliable 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