Workgroup: Transport Area Working Group
Internet-Draft:
draft-ietf-tsvwg-multipath-dccp-00
Published: 31 August 2021
Intended Status: Experimental
Expires: 4 March 2022

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

DCCP Extensions for Multipath Operation with Multiple Addresses

Abstract

DCCP communication is currently 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 this 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 flows across potentially disjoint paths.

Status of This Memo

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 is suggested in the context of ongoing 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, re-assembly 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 flows 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 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.

```
                             +-------------------------------+
                             |           Application         |
+---------------+            +-------------------------------+
|  Application  |            |            MP-DCCP            |
+---------------+            + - - - - - - - + - - - - - - - +
|      DCCP     |            |Subflow (DCCP) |Subflow (DCCP) |
+---------------+            +-------------------------------+
|      IP       |            |       IP      |      IP       |
+---------------+            +-------------------------------+
```

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.

(MP-DCCP) Connection: A set of one or more subflows, over which an application can communicate between two hosts. There is a one-to-one mapping between a connection and an application socket.

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

```
           Host A                                Host B
------------------------             ------------------------
Address A1    Address A2             Address B1    Address B2
----------    ----------             ----------    ----------
  |             |                      |             |
  |         (DCCP flow setup)          |             |
  |----------------------------------->|             |
  |<-----------------------------------|             |
  |             |                      |             |
  |             |  (DCCP flow setup)   |             |
  |             |--------------------->|             |
  |             |<---------------------|             |
  | merge individual DCCP flows to one multipath connection
  |             |                      |             |
```

Figure 2: Example MP-DCCP Usage Scenario

1.4. 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 1 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
Flow control	yes	no
Congestion control	yes	yes

Feature	**TCP**	**DCCP**
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 1: TCP and DCCP protocol comparison

Consequently, the multipath features, shown in Table 2, 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
Robust session establishment	no	yes
Data re-assembly	yes	optional / modular
Expandability	limited by TCP header	flexible

Table 2: MPTCP and MP-DCCP protocol comparison

Therefore, the sender logic is not much different between MP-DCCP and MPTCP, even if the multipath session initiation differs. MP-DCCP inherits a robust session establishment feature, which guarantees communication establishment if at least one functional path is available. MPTCP relies on an initial path, which has to work; otherwise no communication can be established.

The receiver side for MP-DCCP has to deal with the unreliable transport character of DCCP and a possible re-assembly of the data stream while not advocating it. As many unreliable applications have built-in application support for reordering (such as adaptive audio and video buffers), those applications might not need support for re-assembly. However, for applications that benefit from partial or full support of reordering, MP-DCCP can provide flexible support for re-assembly, even if for DCCP the order of delivery is unreliable by nature. Such optional re-assembly mechanisms may account for the fact that packet loss may occur for any of the DCCP subflows. Another issue may occur as packet reordering may happen 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.

1.5. 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

RFC 4340 states that some applications might want to share congestion control state among multiple DCCP flows between same source and destination addresses. This functionality could be provided by the Congestion Manager (CM) [RFC3124], a generic multiplexing facility. However, the CM would not fully support MP-DCCP without change; it does not gracefully handle multiple congestion control mechanisms, for example.

The operation of MP-DCCP for data transfer takes one input data stream from an application, and splits it into one or more subflows, with sufficient control information to allow received data to be re-assembled and delivered in order to the recipient application. The following subsections define this behavior in detail.

The Multipath Capability for MP-DCCP can be negotiated with a new DCCP feature, as described 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.

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 3.

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

Number	Meaning	Rule	Rec'n Value	Initial Req'd
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 3: Proposed Feature Set

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 4.

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 4: Proposed Option Set

[Tbd/tbv] In addition to the multipath option, MP-DCCP requires particular considerations for:

*The minimum PMTU of the individual paths must be announced to the application. Changes of individual path PMTUs must be reannounced to the application if they result in a value lower than the currently announced PMTU.

*Overall sequencing for optional re-assembly procedure

*Congestion control

*Robust MP-DCCP session establishment (no dependency on an initial path setup)

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 is server-priority. It takes one-byte values. The first four bits are used to specify compatible versions of the MP-DCCP implementation. The following four bits are reserved for further use.

3.2. Multipath Option

```
+--------+--------+--------+--------+--------
|00101110| Length | MP_OPT | Value(s) ...
+--------+--------+--------+--------+--------
 Type=46
```

Type	Option Length	MP_OPT	Meaning
46	var	0 =MP_CONFIRM	Confirm reception and processing of an MP_OPT option
46	11	1 =MP_JOIN	Join path to an existing MP-DCCP flow
46	3	2 =MP_FAST_CLOSE	Close MP-DCCP flow
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	var	8 =MP_REMOVEADDR	Remove Address
46	4	9 =MP_PRIO	Change Subflow Priority

Table 5: MP_OPT Option Types

3.2.1. MP_CONFIRM

```
+--------+--------+--------+--------+--------+--------+--------+
|00101110| Length |00000000| List of options ...
+--------+--------+--------+--------+--------+--------+--------+
 Type=46           MP_OPT=0
```

MP_CONFIRM can be used to send confirmation of received and processed options. Confirmed options are copied verbatim and appended as List of options. The length varies dependent on the amount of options.

[Tbd] Encoding "list of options"

3.2.2. MP_JOIN

```
+--------+--------+--------+--------+--------+--------+--------+
|00101110|00001011|00000001| Path Token                        |
+--------+--------+--------+--------+--------+--------+--------+
| Nonce                             |
+--------+--------+--------+--------+
 Type=46  Length=11 MP_OPT=1
```

The MP_JOIN option is used to add a new path to an existing MP-DCCP flow. The Path Token is the SHA-1 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.

3.2.3. MP_FAST_CLOSE

```
+--------+--------+--------+
|00101110|00000011|00000010|
+--------+--------+--------+
 Type=46  Length=3 MP_OPT=2
```

MP_FAST_CLOSE terminates the MP-DCCP flow and all corresponding subflows.

3.2.4. MP_KEY

```
+--------+--------+--------+--------+--------+--------+--------+
|00101110| Length |00000011|Key Type| Key Data ...
+--------+--------+--------+--------+--------+--------+--------+
 Type=46           MP_OPT=3
```

The MP_KEY suboption is used to exchange key material between hosts. The Length varies between 5 and 8 Bytes. The Key Type field is used to specify the key type. Key types are shown in Table 6.

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

Table 6: 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=(key-a+key-b), hostB d-key=(key-b+key-a)).

ECDHE-SHA256-C25519

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

ECDHE-SHA512-C25519

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

3.2.5. MP_SEQ

```
+--------+--------+--------+--------+--------+--------+--------+
|00101110|00000111|00000100| Multipath Sequence Number         |
+--------+--------+--------+--------+--------+--------+--------+
 Type=46  Length=7 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 path individual sequence number space.

3.2.6. MP_HMAC

```
+--------+--------+--------+--------+--------+--------+
|00101110|00001011|00000101| HMAC-SHA1 (20 bytes) ...
+--------+--------+--------+--------+--------+--------+
 Type=46  Length=23 MP_OPT=5
```

The MP_HMAC option is used to provide authentication for the MP_JOIN option. The HMAC is built using the derived key (d-key) calculated previously from the handshake key material exchanged with the MP_KEY option. The Message for the HMAC is the header of the MP_JOIN for which authentication shall be performed. By including a nonce in these datagrams, possible replay-attacks are remedied.

3.2.7. MP_RTT

```
+--------+--------+--------+--------+--------+--------+--------+
|00101110|00001100|00000110|RTT Type| RTT
+--------+--------+--------+--------+--------+--------+--------+
|        | Age                               |
+--------+--------+--------+--------+--------+
 Type=46  Length=12 MP_OPT=6
```

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.

Raw RTT (=0)
Raw RTT value of the last Datagram Round-Trip. The Age parameter is set to the age of when the Ack for the datagram was received.

Min RTT (=1)
Min RTT value. The period for computing the Minimum can be specified by the Age parameter.

Max RTT (=2)
Max RTT value. The period for computing the Maximum can be specified by the Age parameter.

Smooth RTT (=3)
Averaged RTT value. The period for computing the smoothed RTT can be specified by the Age parameter.

Age (=4)
The Age parameter is a 4-byte value which is set to the age or timestamp when the Ack for the datagram was received in case of RTT type = 0 and may contain the periods for computing of derived RTT values depending on other RTT types, i.e., the Minimum (=1) and Maximum (=2) as well as the averaged smoothed RTT value (=3). [TBD/TBV]

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. Length is variable depending on IPv4 or IPv6 and whether port number is used and is in range between 28 and 42 bytes.

```
                    1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+-------+-------+---------------+
|     Kind      |     Length    |Subtype| IPVer |  Address ID   |
+---------------+---------------+-------+-------+---------------+
|          Address (IPv4 - 4 bytes / IPv6 - 16 bytes)           |
+-------------------------------+-------------------------------+
|   Port (2 bytes, optional)    |                               |
+-------------------------------+                               |
|                       HMAC (20 bytes)                         |
|                                                               |
|                                                               |
|                                                               |
|                                                               |
|                               +-------------------------------+
|                               |
+-------------------------------+
```

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 ADD_ADDR 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.

Ideally, ADD_ADDR and REMOVE_ADDR options would be sent reliably, and in order, to the other end. This would ensure that this address management does not unnecessarily cause an outage in the connection when remove/add addresses are processed in reverse order, and also to ensure that all possible paths are used. Note, however, that losing reliability and ordering will not break the multipath connections, it will just reduce the opportunity to open new paths and to survive different patterns of path failures.

Therefore, implementing reliability signals for these DCCP options is not necessary. In order to minimize the impact of the loss of these options, however, it is RECOMMENDED that a sender should send these options on all available subflows. If these options need to be received in order, an implementation SHOULD only send one ADD_ADDR/ REMOVE_ADDR option per RTT, to minimize the risk of misordering. A host that receives an ADD_ADDR 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. A sender that wants to trigger a new incoming connection attempt on a previously advertised address/port combination can therefore refresh ADD_ADDR information by sending the option again.

[TBD/TBV]

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 (REMOVE_ADDR) option which will remove a previously added address (or list of addresses) from a connection and terminate any subflows currently using that address.

For security purposes, if a host receives a REMOVE_ADDR option, it must ensure the affected path(s) are no longer in use before it instigates closure. Typical DCCP validity tests on the subflow (e.g., packet type specific sequence and acknowledgement number check) MUST also be undertaken. An implementation can use indications of these test failures as part of intrusion detection or error logging.

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.

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+---------------+---------------+-------+-------+---------------+
|     Kind      |  Length = 3+n |Subtype|(resvd)|   Address ID  |...
+---------------+---------------+-------+-------+---------------+
                          (followed by n-1 Address IDs, if required)
```

Minimum length of this option is 4 bytes (for one address to remove).

[TBD/TBV]

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, a host may wish to signal such change in priority of subflows to the peer. Therefore, the MP_PRIO option, shown below, can be used to set a priority flag for the subflow on which it is sent.

```
                       1                   2                   3
   0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
  +---------------+---------------+-------+-------+---------------+
  |     Kind      |     Length    |Subtype| Prio  | AddrID (opt)  |
  +---------------+---------------+-------+-------+---------------+
```

Whether more than two values for priority (e.g., B for backup and P for prioritized path) are defined in case of more than two parallel paths is for further consideration.

[TBD/TBV]

3.3. MP-DCCP Handshaking Procedure

```
          Host A                                         Host B
------------------------                              ----------
Address A1    Address A2                              Address B1
----------    ----------                              ----------
     |             |                                       |
     |             DCCP-Request +                          |
     |------- MP_KEY(Key-A) ------------------------------>|
     |<---------------------- MP_KEY(Key-B) ---------------|
     |             DCCP-Response +  agreed                 |
     |             |                                       |
     |   DCCP-Ack  |                                       |
     |--------- MP_KEY(Key-A) + MP_KEY(Key-B) ------------>|
     |             |                                       |
     |             |          DCCP-Request +               |
     |             |--- MP_JOIN(TB,RA) ------------------->|
     |             |<------MP_JOIN(TB,RB) + MP_HMAC(A)-----|
     |             |DCCP-Response                          |
     |             |                                       |
     |             |DCCP-Ack                               |
     |             |-------- MP_HMAC(B) ------------------>|
     |             |<--------------------------------------|
     |             |DCCP-ACK                               |
```

Figure 3: Example MP-DCCP Handshake

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

- *Host A sends a DCCP-Request with the MP-Capable feature Change request and the MP_KEY option with Host-specific Key-A

- *Host B sends a DCCP-Response with Confirm feature for MP-Capable and the MP_Key option with Host-specific Key-B

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

The handshake for subsequent flows 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 SHA-1 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.

- *Host A sends a DCCP-Ack with the HMAC computed for the DCCP-Response.

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

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 of today. 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 today.

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

1. Notes

This document is inspired by Multipath TCP [RFC6824]/[RFC8684] and some text passages for the -00 version of the draft are copied almost unmodified.

8. IANA Considerations

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 7 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 7: 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 8 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 path to existing MP-DCCP flow	Section 3.2.2
TBD or MP_OPT=2	MP_FAST_CLOSE	Close MP-DCCP flow	Section 3.2.3
TBD or MP_OPT=3	MP_KEY	Exchange key material for MP_HMAC	Section 3.2.4
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

Table 8: Addition to DCCP Protocol options and corresponding sub-options

[Tbd], must include options for:

*handshaking procedure to indicate MP support

*handshaking procedure to indicate JOINING of an existing MP connection

*signaling of new or changed addresses

*setting handover or aggregation mode

*setting reordering on/off

should include options carrying:

*overall sequence number for restoring purposes

*sender time measurements for restoring purposes

*scheduler preferences

*reordering preferences

9. Informative References

[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>.

[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>.

[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>.

[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>.

[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/>.

Authors' Addresses

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

Email: Markus.Amend@telekom.de

Dirk von Hugo
Deutsche Telekom
Deutsche-Telekom-Allee 9
64295 Darmstadt
Germany

Email: Dirk.von-Hugo@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