Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2017

M. Boucadair, Ed.
C. Jacquenet, Ed.
Orange
O. Bonaventure, Ed.
Tessares
D. Behaghel
OneAccess
S. Secci
UPMC
W. Henderickx, Ed.
Nokia/Alcatel-Lucent
R. Skog, Ed.
Ericsson
S. Vinapamula
Juniper
S. Seo
Korea Telecom
W. Cloetens
SoftAtHome
U. Meyer
Vodafone
LM. Contreras
Telefonica
B. Peirens
Proximus

October 27, 2016

An MPTCP Option for Network-Assisted MPTCP

draft-boucadair-mptcp-plain-mode-09

Abstract

Because of the lack of Multipath TCP (MPTCP) support at the server side, some service providers now consider a network-assisted model that relies upon the activation of a dedicated function called MPTCP Conversion Point (MCP). Network-Assisted MPTCP deployment models are designed to facilitate the adoption of MPTCP for the establishment of multi-path communications without making any assumption about the support of MPTCP by the communicating peers. MCPs located in the network are responsible for establishing multi-path communications on behalf of endpoints, thereby taking advantage of MPTCP capabilities to achieve different goals that include (but are not limited to) optimization of resource usage (e.g., bandwidth aggregation), of resiliency (e.g., primary/backup communication paths), and traffic offload management.

This document specifies an MPTCP option that is used in the context of Network-Assisted MPTCP: MP_CONVERT.

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 RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft will expire on April 30, 2017.

Copyright Notice

Table of Contents

1. Introduction

The overall quality of connectivity services can be enhanced by combining several access network links for various purposes - resource optimization, better resiliency, etc. Some transport protocols, such as Multipath TCP [RFC6824], can help achieve such better quality, but failed to be massively deployed so far.

The support of multipath transport capabilities by communicating hosts remains a privileged target design so that such hosts can directly use the available resources provided by a variety of access networks they can connect to. Nevertheless, network operators do not control end hosts while the support of MPTCP by content servers remains close to zero.

Network-Assisted MPTCP deployment models are designed to facilitate the adoption of MPTCP for the establishment of multi-path communications without making any assumption about the support of MPTCP capabilities by communicating peers. Network-Assisted MPTCP deployment models rely upon MPTCP Conversion Points (MCPs) that act

on behalf of hosts so that they can take advantage of establishing communications over multiple paths. MCPs can be deployed in CPEs (Customer Premises Equipment), as well as in the provider's network. MCPs are responsible for establishing multi-path communications on behalf of endpoints. Further details about the target use cases are provided in Section 3.

Most of the current operational deployments that take advantage of multi-interfaced devices rely upon the use of an encapsulation scheme (such as [I-D.zhang-gre-tunnel-bonding], [TR-348]). The use of encapsulation is motivated by the need to steer traffic towards the concentrator and also to allow the distribution of any kind of traffic besides TCP (e.g., UDP) among the available paths without requiring any advanced traffic engineering tweaking technique in the network to intercept traffic and redirect it towards the appropriate MCP.

Current operational MPTCP deployments by network operators are focused on the forwarding of TCP traffic. The design of such deployments sometimes assumes the use of extra signalling provided by SOCKS [RFC1928], at the cost of additional management complexity and possible service degradation (e.g., up to 8 SOCKS messages may have to be exchanged between two MCPs before an MPTCP connection is established, thereby yielding several tens of milliseconds of extra delay before the connection is established) .

To avoid the burden of encapsulation and additional signalling between MCPs, this document explains how a plain transport mode is enabled, so that packets are exchanged between a device and its upstream MCP without requiring the activation of any encapsulation scheme (e.g., IP-in-IP [RFC2473], GRE [RFC1701]). This plain transport mode also avoids the need for out-of-band signalling, unlike the aforementioned SOCKS context.

The solution described in this document also works properly when NATs are present in the communication path between a device and its upstream MCP. In particular, the solution in this document accommodates deployments that involve CGN (Carrier Grade NAT) upstream the MCP.

Network-Assisted MPTCP deployment and operational considerations are discussed in [I-D.nam-mptcp-deployment-considerations].

The plain transport mode is characterized as follows:

- No encapsulation required (no tunnels, whatsoever).
- No out-of-band signaling for each MPTCP subflow is required.
- Carries any protocol for the benefit of massive MPTCP adoption.

- o Avoids interference with native MPTCP connections.
- o Targets both on-path and off-path MCPs.
- o Accommodates various deployment contexts, such as those that require the preservation of the source IP address and others characterized by an address sharing design.

2. Terminology

The reader should be familiar with the terminology defined in [RFC6824].

This document makes use of the following terms:

- o Client: an endhost that initiates transport flows forwarded along a single path. Such endhost is not assumed to support multipath transport capabilities.

- o Server: an endhost that communicates with a client. Such endhost is not assumed to support multipath transport capabilities.

- o Multipath Client: a Client that supports multipath transport capabilities.

- o Multipath Server: a Server that supports multipath transport capabilities. Both the client and the server can be single-homed or multi-homed. However, for the use cases discussed in this document, the number of interfaces available at the endhosts is not relevant.

- o Transport flow: a sequence of packets that belong to a unidirectional transport flow and which share at least one common characteristic (e.g., the same destination address). TCP and SCTP flows are composed of packets that have the same source and destination addresses, the same protocol number and the same source and destination ports.

- o Multipath Conversion Point (MCP): a function that terminates a transport flow and relays all data carried in the flow into another transport flow.

  MCP is a function that converts a multipath transport flow and relays it over a single path transport flow and vice versa.

3. Target Use Cases

We consider two important use cases in this document. We briefly introduce them in this section and leave the details to Section 5 and Section 6. The first use case is a Multipath Client that interacts

with a remote Server through a MCP (Section 3.1). The second use case is a multi-homed CPE that includes a MCP and interacts with a remote Server through a downstream MCP (Section 3.2).

3.1. Multipath Client

In this use case, the Multipath Client would like to take advantage of MPTCP even if the Server does not support MPTCP. A typical example is a smartphone that could use both WLAN and LTE access networks to reach a Server in order to achieve higher bandwidth or better resilience.

```
+--+                                      +------+      +--+
|H1|                                      | MCP  |      |RM|
+--+                                      +------+      +--+
 |                                            |            |
 |<==================MPTCP Leg==============>|<---TCP -->|
 |                                            |            |

Legend:
    H1: Host 1
   MCP: Multipath Conversion Point
    RM: Remote Machine
```

Figure 1: Network-assisted MPTCP (Host-based Model)

In reference to Figure 1, the MCP terminates the MPTCP connection established by the Client and binds it to a TCP connection towards the remote Server. Two deployments of this use case are possible.

A first deployment is when the MCP is on the path between the Multipath Client and the Server. In this case, the MCP can terminate the MPTCP connection initiated by the Client and binds it to a TCP connection that the MCP establishes with the Server. Because the MCP is not located on all default forwarding paths, the MPTCP connection must be initiated by using the path where the MCP is located.

A second deployment is when the MCP is not on the path between the Multipath Client and the Server. In this case, the Client must first initiate a connection towards the MCP and request it to initiate a TCP connection towards the Server. This is what the SOCKS protocol performs by exchanging control messages to create appropriate mappings to handle the connection. Unfortunately, this requires additional round-trip-time that affects the performance of the end-to-end data transfer, in particular for short-lived connections. This document proposes the MP_CONVERT option which is carried in the SYN segment of the initial subflow. This SYN segment is sent towards the MCP. The MP_CONVERT option contains the destination address (and

optionally a port number) of the Server. Thanks to this information, the MCP can immediately establish the TCP connection with the Server without any additional round-trip-time, unlike a SOCKS-based design.

3.2. Multipath CPE

In this use case, neither the Client nor the Server support MPTCP. Two MCPs are used as illustrated in Figure 2. The upstream MCP is embedded in the CPE while the downstream MCP is located in the provider's network. The CPE is attached to multiple access networks (e.g., xDSL and LTE). The upstream MCP transparently terminates the TCP connections initiated by the Client and converts them into MPTCP connections.

```
             Upstream                      Downstream
     +--+      +-----+                       +-----+      +--+
     |H1|      | MCP |                       | MCP |      |RM|
     +--+      +-----+                       +-----+      +--+
      |           |                             |           |
      |<---TCP--->|<========MPTCP Leg==========>|<---TCP--->|
      |           |                             |           |
```

Figure 2: Network-assisted MPTCP (CPE-based Model)

The same considerations detailed in Section 3.1 apply for the insertion of the downstream MCP in an MPTCP connection.

4. MP_CONVERT Option

The MP_CONVERT (MC) option carries the source/destination IP addresses and/or port numbers of the origin source and destination nodes. It is also used to indicate whether the data carried in the packet is relayed from a native TCP connection or refers to the use of another transport protocol.

The format of the option is shown in Figure 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|D|Flags|    Protocol   |
+---------------+---------------+-------+-+-----+---------------+
|          Address (IPv4 - 4 octets / IPv6 - 16 octets)         |
+-------------------------------+-------------------------------+
|   Port (2 octets, optional)   |
+-------------------------------+
```

Figure 3: MP_CONVERT Option

The description of the fields is as follows:

- Kind and Length: are the same as those defined in Section 3 of [RFC6824]. The minimum size of this option is 4 bytes.

- Subtype: to be defined by IANA (Section 8). Implementations may use the "0xe" subtype encoding for early deployment purposes in managed networks.

- D-bit (Direction bit): this flag indicates whether the enclosed IP address (and port number) reflects the source or the destination IP address (and port number). When the D-bit is set, the enclosed IP address must be interpreted as the source IP address. When the D-bit is unset, the enclosed IP address must be interpreted as the destination IP address.

- "Flags" bits: are reserved bits for future assignment as additional flag bits. These additional flag bits MUST each be set to zero and MUST be ignored upon receipt.

- Protocol: conveys the protocol number as assigned by IANA [proto_numbers].

- Address: includes a source or destination IP address. The address family is determined by the "Length" field. Concretely, a MP_CONVERT option that carries an IPv4 address has a Length field of 8 bytes (or 10, if a port number is included). A MP_CONVERT option that carries an IPv6 address has a Length of 20 bytes (or 22, if a port number is included). This field is optional.

- Port: If the D-bit is set (resp. unset), a source (resp. destination) port number may be associated with the IP address. This field is valid for protocols that use a 16 bit port number (e.g., UDP, TCP, SCTP). This field is optional.

The MP_CONVERT option is a variable length MPTCP option that MUST NOT be used in TCP segments whose SYN flag is reset. This option can only appear in the SYN used to create the initial subflow of a Multipath TCP connection (see the example in Figure 4).

Up to two MP_CONVERT options can appear inside a SYN segment. If two MP_CONVERT options are included, these options MUST NOT have the same D-bit value.

```
        +----+                              +-----+       +--+
        |  C |                              | MCP |       |S |
        +----+                              +-----+       +--+
         |  | ________________________________|             |
         |  /        Initial subflow           \            |
         |  |========SYN(MP_CAPABLE+MC(S))===>|             |
         |  |                                 |--SYN------->|
         |  |                                 |<--SYN/ACK---|
         |  |<====SYN/ACK(MP_CAPABLE)=========|             |
         |  |             ...                 |             |
         |  \ ________________________________/             |
                         ....                      ....
         |  | ________________________________|             |
         |  /        Additional subflow        \            |
         |  \ ________________________________/             |

        Legend:
             <===>: MPTCP leg
             <--->: TCP leg
```

Figure 4: Carrying the MP_CONVERT Option

The MP_CONVERT option MUST be included in the SYN payload.

NOTE 1: Given the length of the MP_CONVERT option, especially when IPv6 addresses are used, and the set of TCP options that are likely to be included in a SYN message, it will not always be possible to place the MP_CONVERT option inside the dedicated TCP option space.

NOTE 2: Including data in a SYN payload is allowed as per Section 3.4 of [RFC0793].

NOTE 3: Stateless approaches that rely on inserting the MP_CONVERT option in all packets are out of scope.

DISCUSSION NOTE: ADD DETAILS ABOUT THE NEED FOR AN EXPLICIT SIGNAL THAT MPTCP OPTIONS ARE INCLUDED IN THE SYN PAYLOAD?

If the MP_CONVERT option appears in either a SYN segment that does not include the MP_CAPABLE option or a segment whose SYN flag is reset, it MUST be ignored. An implementation MAY log this event since it likely indicates an operational issue.

If the original SYN message contains data in its payload (e.g., [RFC7413]), that data MUST be placed right after the MP_CONVERT and "End of Options List" (EOL) options when generating the SYN in the MPTCP leg. The EOL option serves as a marker to delineate the end of

the TCP options and the beginning of the data included in the original SYN .

An implementation MUST ignore MP_CONVERT options that include multicast, broadcast, and host loopback addresses [RFC6890]. Concretely, an implementation that receives an MP_CONVERT option with such addresses MUST silently tear down the MPTCP connection.

When an MCP creates an MPTCP connection triggered by an incoming packet, it MUST copy in the 'Protocol' field of the MP_CONVERT option the value of the 'Protocol' field (resp. type of the transport header) of the IPv4 (resp. IPv6) header of this incoming packet. The MCP may be configured to enable traffic aggregation for some transport protocols because of the nature of the service they relate to. By default, the 'Protocol' field MUST be set to 6 (TCP).

5. MPTCP Connections from a Multipath TCP Client

5.1. Description

The simplest usage of the MP_CONVERT option is when a Multipath TCP Client wants to use MPTCP to efficiently utilise different network paths (e.g., WLAN and LTE from a smartphone) to reach a server that does not support Multipath TCP. The basic operation is illustrated in Figure 5.

To use its multipath capabilities to establish an MPTCP connection over the available networks, the Client splits its end-to-end connection towards the TCP Server into two:

(1) An MPTCP connection, that typically relies upon the establishment of one subflow per network path, is established between the client and the MCP.

(2) A TCP connection that is established by the MCP with the server.

Any data that is eligible to be transported over the MPTCP connection is sent by the Client towards the MCP over the MPTCP connection. The MCP then forwards these data over the regular TCP connection until they reach the server. The same forwarding principle applies for the data sent by the Server over the TCP connection with the MCP.

```
        C <===========>MCP <------------> S
        +<=============>+

   Legend:
     <===>: subflows of the upstream MPTCP connection
     <--->: downstream TCP connection
```

Figure 5: A Multipath TCP Client interacts with a Server through a Multipath Conversion Point

5.2. Theory of Operation

We assume in this section that the Multipath TCP Client has been configured with the IP address of one or more MCPs which convert the Multipath TCP connection into a regular TCP connection. The address of such MCPs can be statically configured on the Client, dynamically provisioned to the MPTCP Client by means of a DHCP option [I-D.boucadair-mptcp-dhc] or by any other means that are outside the scope of this document.

Conceptually, the MCP acts as a relay between an upstream MPTCP connection and a downstream TCP connection. The MCP has at least a single IP address that is reachable from the Multipath TCP Client. It may be assigned other IP addresses. For the sake of simplicity, we assume in this section that the MCP has a single IP address denoted MCP@. Similarly, we assume that the client has two addresses C@1 and C@2 while address S@ is assigned to the server.

The MCP maps an upstream MPTCP connection (and its associated subflows) onto a downstream TCP connection. On the MCP, an established Multipath TCP connection can be identified by the local Token that was assigned upon reception of the SYN segment.

This Token is guaranteed to be unique on the MCP (provided that it has a single IP address) during the entire lifetime of the MPTCP connection. The 4-tuple (IP src, IP dst, Port src, Port dst) is used to identify the downstream TCP connection.

To initiate a connection to a remote server S, the Multipath TCP Client sends a SYN segment towards the MCP that includes the MP_CONVERT option described in Figure 3. The destination address of the SYN segment is the IP address of the MCP. The MP_CONVERT option included in the SYN contains the IP address and optionally the destination port of the Server (see Figure 6).

```
          +----+                               +-----+    +--+
          |  C |                               | MCP |    |S |
          +----+                               +-----+    +--+
          C@1 C@2                               MCP@       S@
           |  | ________________________________|          |
           |  /       Initial subflow          \          |
           |  |=======SYN(MP_CAPABLE+MC(S@))===>|          |
           |  |                                |--SYN---->|
           |  |                                |<-SYN/ACK-|
           |  |<====SYN/ACK(MP_CAPABLE)=========|          |
           |  |             ...                |          |
           |  \ ________________________________/          |
                         ....                     ....
           |  | ________________________________|          |
           |  /       Additional subflow       \          |
           |  \ ________________________________/          |

          Legend:
               <===>: MPTCP leg
               <--->: TCP leg
```

Figure 6: Single-ended MCP Flow Example

The MCP processes this SYN segment as follows. First, it generates the local key and a unique Token for the Multipath TCP connection. This Token identifies the MPTCP connection. It is passed to the MCP together with the contents of the MP_CONVERT option (i.e., the address of the destination server) and the destination port.

The MCP then establishes a TCP connection with the destination server. If the received MP_CONVERT option contains a port number, it is used as the destination port of the outgoing TCP connection that is being established by the MCP. Otherwise, the destination port of the upstream MPTCP connection is used as the destination port of the downstream TCP connection. The MCP creates a flow entry for the downstream TCP connection and maps the upstream MPTCP connection onto the downstream TCP connection.

The downstream TCP connection is considered to be active upon reception of the SYN+ACK segment sent by the destination server. The reception of this segment triggers the MCP that confirms the establishment of the upstream MPTCP connection by sending a SYN+ACK segment towards the Multipath TCP Client.

At this point, there are two established connections. The endpoints of the upstream Multipath TCP connection are the Multipath TCP Client and the MCP. The endpoints of the downstream TCP connection are the MCP and the Server. These two connections are bound by the MCP.

All the techniques defined in [RFC6824] can be used by the upstream Multipath TCP connection. In particular, the subflows established over the different network paths can be controlled by either the Multipath TCP Client or the MCP. It is likely that the network operators that deploy MCPs will define policies for the utilisation of the MCP. These policies are discussed in [I-D.nam-mptcp-deployment-considerations].

Any data received by the MCP on the upstream Multipath TCP connection will be forwarded by the MCP over the bound downstream TCP connection. The same applies for data received over the downstream TCP connection which will be forwarded by the MCP over the upstream Multipath TCP connection.

One of the functions of the MCP is to maintain the binding between the upstream Multipath TCP connection and the downstream TCP connection. If the downstream TCP connection fails for some reason (excessive retransmissions, reception of a RST segment, etc.), then the MCP SHOULD force the teardown of the upstream Multipath TCP connection by transmitting a FASTCLOSE. Similarly, if the upstream Multipath TCP connection fails for some reason (e.g., reception of a FASTCLOSE), the MCP SHOULD tear the downstream TCP connection down and remove the flow entries.

The same reasoning applies when the upstream Multipath TCP connection ends with the transmission of DATA_FINs. In this case, the MCP SHOULD also terminate the bound downstream TCP connection by using FIN segments. If the downstream TCP connection terminates with the exchange of FIN segments, the MCP SHOULD initiate a graceful termination of the bound upstream Multipath TCP connection.

An MCP SHOULD associate a lifetime with the Multipath TCP and TCP flow entries. In this case, it SHOULD use the same lifetime for each pair of bounded connections.

6. MPTCP Connections Between Single Path Client and Server

6.1. Description

There are situations where neither the client nor the server can use multipath transport protocols albeit network providers would want to optimize network resource usage by means of multi-path communication techniques. Hybrid access service offerings are typical business incentives for such situations, where network operators combine a fixed network (e.g., xDSL) with a wireless network (e.g., LTE). In this case, as illustrated in Figure 7, two MCPs are used for each flow. The first MCP, located downstream of the client, converts the single path TCP connection originated from the client into a

Multipath TCP connection established with a second MCP. The latter will then establish a TCP connection with the destination server.

```
          Upstream        Downstream
     C <---> MCP <===========> MCP <------------> S
               +<=============>+
```

Legend:
     <===>: MPTCP leg
     <--->: TCP leg

Figure 7: A Client interacts with a Server through an upstream and a downstream Multipath Conversion Points

6.2. Theory of Operation

6.2.1. Downstream MCP

The downstream MCP can be deployed on-path or off-path. If the downstream MCP is deployed off-path, its behavior is described in Section 5.2.

If the downstream MCP is deployed on-path, it only terminates MPTCP connections that carry an empty MP_CONVERT option inside their SYN (i.e., no address is conveyed). If the MCP receives a SYN segment that contains the MP_CAPABLE option but no MP_CONVERT option, it MUST forward the SYN to its final destination without any modification.

6.2.2. Upstream MCP

The upstream and downstream MCPs cooperate. The upstream MCP may be configured with the addresses of downstream MCPs. If the downstream MCP is deployed on-path, the upstream MCP inserts an MP_CONVERT option that carries no IP address.

In this section, we assume that the upstream MCP has been configured with one address of the downstream MCP. This address can be configured statically, dynamically distributed by means of a DHCP option [I-D.boucadair-mptcp-dhc] or by any other means that are outside the scope of this document.

We assume that the upstream MCP has two addresses uMCP@1 and uMCP@2 while the downstream MCP is assigned a single IP address dMCP@.

The upstream MCP maps an upstream TCP connection onto a downstream MPTCP connection (and its associated subflows) . On the upstream MCP, an established MPTCP connection can be identified by the local Token that was assigned upon reception of the SYN segment from the Client.

To initiate a connection with a remote server S, the Client sends a SYN segment that is intercepted by the upstream MCP which in turns initiates an MPTCP connection towards its downstream MCP that includes the MP_CONVERT option described in Figure 3. The destination address of the SYN segment is the IP address of the downstream MCP. The MP_CONVERT option included in the SYN contains the IP address and optionally the destination port of the Server; this information is extracted from the SYN message received over the upstream TCP connection.

Concretely, the upstream MCP processes the SYN segment received from the Client as follows.

First, it generates the local key and a unique Token for the Multipath TCP connection to identify the MPTCP connection. It extracts the destination IP address and, optionally, the destination port that will then be carried in a MP_CONVERT option. The upstream MCP establishes an MPTCP connection with the downstream MCP. The upstream MCP creates a flow entry for the downstream MPTCP connection and maps the upstream TCP connection onto the downstream MPTCP connection.

The downstream MPTCP connection is considered to be active upon reception of the SYN+ACK segment from the downstream MCP. The reception of this segment triggers the upstream MCP that confirms the establishment of the upstream TCP connection by sending a SYN+ACK segment towards the TCP Client.

At this point, there are two established connections maintained by the upstream MCP:

(1) The endpoints of the upstream TCP connection are the Client and the upstream MCP.

(2) The endpoints of the downstream MPTCP connection are the upstream MCP and the downstream MCP.

These two connections are bound by the upstream MCP. An example is shown in Figure 8.

```
             Upstream                        Downstream
   +--+      +-----+                          +-----+      +--+
   |C1|      | MCP |                          | MCP |      |S1|
   +--+      +-----+                          +-----+      +--+
    C@1   uMCP@1 uMCP@2                         dMCP@       S@
     |          | |______________________________|          |
     |--SYN--->|/       Initial subflow          \          |
     |          |=======SYN(MP_CAPABLE+MC(S@))==>|          |
     |          |                                |--SYN---->|
     |          |                                |<-SYN/ACK-|
     |          |<====SYN/ACK(MP_CAPABLE)========|          |
     |<SYN/ACK-|              ...                |          |
     |          \ ______________________________/           |
                           ....                    ....
     |          | | ____________________________ |          |
     |          | |/       Additional subflow   \|          |
     |          | |\ ___________________________/|          |
                               ....
```

Figure 8: Dual-Ended MCP Flow Example

All the techniques defined in [RFC6824] can be used by the MPTCP connection. In particular, the utilisation of the different network paths can be controlled by one MCP or the other.

Any data received by the upstream MCP over the upstream TCP connection will be forwarded by the MCP over the bound downstream MPTCP connection, assuming such data are eligible to MPTCP transport. The same applies for data received over the downstream MPTCP connection which will be forwarded by the upstream MCP over the upstream TCP connection.

The same considerations as in Section 5.2 apply for the maintenance of the connections by the upstream MCP.

7. Demux Native MPTCP Connections From Proxied MPTCP Connections

Section 6 assumes that the Clients that use the upstream MCP do not support MPTCP. If a Multipath Client is attached to the upstream MCP, it should be possible for this client to establish an MPTCP connection with a Multipath Server without using the MCPs.

Because MPTCP connections are not destined explicitly to an MCP, an on-path MCP instance will need extra means to distinguish "native" MPTCP connections from "proxied" ones. The subsequent risk is that native MPTCP communications will be reverted to TCP connections as shown in Figure 9. In this example, we suppose that C2 and S2 are MPTCP-compatible, but C1 and S1 are not.

```
          Upstream                     Downstream
+--+      +-----+                         +-----+      +--+
|C1|      | MCP |                         | MCP |      |S1|
+--+      +-----+                         +-----+      +--+
 |           |                               |           |
 |src:c1@    |src:uMCP@1            dst:s1@|src:uMCP@1 |
 |<-TCP Leg->|<========MPTCP Leg===========>|<-TCP Leg->|
 |    dst:s1@|                               |    dst:s1@|
 |           |                               |           |

\_________________SINGLE PATH CLIENT/SERVER_____________/
 _________________MULTIPLE PATH CLIENT/SERVER___________
/                                                        \
             |                               |
 |           |                               |           |
 |src:c2@    |src:uMCP@1            dst:s1@|src:uMCP@1 |
 |<=MPTCP====|==============================>|<-TCP Leg->|
 |    dst:s2@|                               |    dst:s2@|
 |           |                               |           |
+--+      +-----+                         +-----+      +--+
|C2|      | MCP |                         | MCP |      |S2|
+--+      +-----+                         +-----+      +--+
          Upstream                     Downstream
```

Figure 9: Example of a Broken E2E MPTCP Connection (On-path)

To mitigate this, the upstream MCP may be instructed to insert a MP_CONVERT option only for the MPTCP connections it establishes. The absence of MP_CONVERT option instances is an explicit indication that this MPTCP connection is a native one. As such, an on-path MCP will not revert this connection into a TCP connection, but will forward packets without any modification to the next hop.

Figure 10 illustrates the results of such procedure: native MPTCP connections are established between MPTCP-compliant client and server, while Networok-Assisted MPTCP connections are established with the help of MCPs.

Concretely, if the upstream MCP receives a SYN that includes the MP_CAPABLE option, it MAY decide to forward it towards its final destination without modifying it. When the downstream MCP receives a SYN that does not include an MP_CONVERT option, it forwards it towards its final destination.

```
             Upstream                              Downstream
   +--+      +-----+                                +-----+      +--+
   |C1|      | MCP |                                | MCP |      |S1|
   +--+      +-----+                                +-----+      +--+
    |           |                               |           |
    |src:c1@    |src:uMCP@1             dst:s1@|src:uMCP@1 |
    |<-TCP Leg->|<========MPTCP Leg(MC)=======>|<-TCP Leg->|
    |    dst:s1@|                               |    dst:s1@|
    |           |                               |           |

   \_________________SINGLE PATH CLIENT/SERVER_____________/
    _________________MULTIPLE PATH CLIENT/SERVER___________
   /                                                        \
                |                               |
    |           |                               |           |
    |src:c2@    |src:uMCP@1             dst:s1@|src:uMCP@1 |
    |<==========|============MPTCP=======================>|
    |    dst:s2@|                               |    dst:s2@|
    |           |                               |           |
   +--+      +-----+                                +-----+      +--+
   |C2|      | MCP |                                | MCP |      |S2|
   +--+      +-----+                                +-----+      +--+
             Upstream                             Downstream
```

Figure 10: Example of a Successful E2E MPTCP Connection (On-path)

8. IANA Considerations

This document requests an MPTCP subtype code for this option:

- o MP_CONVERT option

  NOTE: Implementations may use "0xe" subtype encoding for early deployment purposes in managed networks.

9. Security Considerations

MPTCP-related security threats are discussed in [RFC6181] and [RFC6824]. Additional considerations are discussed in the following sub-sections.

9.1. Privacy

The MCP may have access to privacy-related information (e.g., IMSI, link identifier, subscriber credentials, etc.). The MCP MUST NOT leak such sensitive information outside a local domain.

9.2. Denial-of-Service (DoS)

Means to protect the MCP against Denial-of-Service (DoS) attacks MUST be enabled. Such means include the enforcement of ingress filtering policies at the network boundaries [RFC2827].

In order to prevent the exhaustion of MCP resources by establishing a great number of simultaneous subflows for each MPTCP connection, the MCP administrator SHOULD limit the number of allowed subflows per CPE for a given connection. Means to protect against SYN flooding attacks MUST also be enabled ([RFC4987]).

Attacks that originate outside of the domain can be prevented if ingress filtering policies are enforced. Nevertheless, attacks from within the network between a host and an MCP instance are yet another actual threat. Means to ensure that illegitimate nodes cannot connect to a network should be implemented.

9.3. Illegitimate MCP

Traffic theft is a risk if an illegitimate MCP is inserted in the path. Indeed, inserting an illegitimate MCP in the forwarding path allows traffic intercept and can therefore provide access to sensitive data issued by or destined to a host. To mitigate this threat, secure means to discover an MCP should be enabled.

10. Acknowledgements

Many thanks to Chi Dung Phung, Mingui Zhang, Rao Shoaib, Yoshifumi Nishida, and Christoph Paasch for their valuable comments.

Thanks to Ian Farrer, Mikael Abrahamsson, Alan Ford, Dan Wing, and Sri Gundavelli for the fruitful discussions in IETF#95 (Buenos Aires).

Special thanks to Pierrick Seite, Yannick Le Goff, Fred Klamm, and Xavier Grall for their inputs.

Thanks also to Olaf Schleusing, Martin Gysi, Thomas Zasowski, Andreas Burkhard, Silka Simmen, Sandro Berger, Michael Melloul, Jean-Yves Flahaut, Adrien Desportes, Gregory Detal, Benjamin David, Arun Srinivasan, and Raghavendra Mallya for the discussion.

11. References

11.1. Normative References

[proto_numbers]
http://www.iana.org/assignments/protocol-numbers, "Protocol Numbers".

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, DOI 10.17487/RFC0793, September 1981, <http://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, <http://www.rfc-editor.org/info/rfc2119>.

[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, <http://www.rfc-editor.org/info/rfc6824>.

[RFC6890] Cotton, M., Vegoda, L., Bonica, R., Ed., and B. Haberman, "Special-Purpose IP Address Registries", BCP 153, RFC 6890, DOI 10.17487/RFC6890, April 2013, <http://www.rfc-editor.org/info/rfc6890>.

11.2. Informative References

[I-D.boucadair-mptcp-dhc]
Boucadair, M., Jacquenet, C., and T. Reddy, "DHCP Options for Network-Assisted Multipath TCP (MPTCP)", draft-boucadair-mptcp-dhc-06 (work in progress), October 2016.

[I-D.nam-mptcp-deployment-considerations]
Boucadair, M., Jacquenet, C., Bonaventure, O., Behaghel, D., stefano.secci@lip6.fr, s., Henderickx, W., Skog, R., Vinapamula, S., Seo, S., Cloetens, W., Meyer, U., Contreras, L., and B. Peirens, "Network-Assisted MPTCP: Use Cases, Deployment Scenarios and Operational Considerations", draft-nam-mptcp-deployment-considerations-00 (work in progress), October 2016.

[I-D.zhang-gre-tunnel-bonding]
Leymann, N., Heidemann, C., Zhang, M., Sarikaya, B., and M. Cullen, "Huawei's GRE Tunnel Bonding Protocol", draft-zhang-gre-tunnel-bonding-03 (work in progress), May 2016.

[RFC1701] Hanks, S., Li, T., Farinacci, D., and P. Traina, "Generic Routing Encapsulation (GRE)", RFC 1701, DOI 10.17487/RFC1701, October 1994, <http://www.rfc-editor.org/info/rfc1701>.

[RFC1928] Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and L. Jones, "SOCKS Protocol Version 5", RFC 1928, DOI 10.17487/RFC1928, March 1996, <http://www.rfc-editor.org/info/rfc1928>.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <http://www.rfc-editor.org/info/rfc2473>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <http://www.rfc-editor.org/info/rfc2827>.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common Mitigations", RFC 4987, DOI 10.17487/RFC4987, August 2007, <http://www.rfc-editor.org/info/rfc4987>.

[RFC6181] Bagnulo, M., "Threat Analysis for TCP Extensions for Multipath Operation with Multiple Addresses", RFC 6181, DOI 10.17487/RFC6181, March 2011, <http://www.rfc-editor.org/info/rfc6181>.

[RFC7413] Cheng, Y., Chu, J., Radhakrishnan, S., and A. Jain, "TCP Fast Open", RFC 7413, DOI 10.17487/RFC7413, December 2014, <http://www.rfc-editor.org/info/rfc7413>.

[TR-348] BBF, "Hybrid Access Broadband Network Architecture", July 2016.

Authors' Addresses

Mohamed Boucadair (editor)
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Christian Jacquenet (editor)
Orange
Rennes
France

Email: christian.jacquenet@orange.com

Olivier Bonaventure (editor)
Tessares
Belgium

Email: olivier.bonaventure@tessares.net

Denis Behaghel
OneAccess

Email: Denis.Behaghel@oneaccess-net.com

Stefano Secci
UPMC

Email: stefano.secci@lip6.fr

Wim Henderickx (editor)
Nokia/Alcatel-Lucent
Belgium

Email: wim.henderickx@alcatel-lucent.com

Robert Skog (editor)
Ericsson

Email: robert.skog@ericsson.com

Suresh Vinapamula
Juniper
1137 Innovation Way
Sunnyvale, CA 94089
USA

Email: Sureshk@juniper.net

SungHoon Seo
Korea Telecom
Seoul
Korea

Email: sh.seo@kt.com

Wouter Cloetens
SoftAtHome
Vaartdijk 3 701
3018 Wijgmaal
Belgium

Email: wouter.cloetens@softathome.com

Ullrich Meyer
Vodafone
Germany

Email: ullrich.meyer@vodafone.com

Luis M. Contreras
Telefonica
Spain

Email: luismiguel.contrerasmurillo@telefonica.com

Bart Peirens
Proximus

Email: bart.peirens@proximus.com