MPTCP Working Group
Internet-Draft
Intended status: Experimental
Expires: January 9, 2017

F. Duchene
O. Bonaventure
UCLouvain

July 08, 2016

Multipath TCP Address Advertisement
draft-duchene-mptcp-add-addr-00

Abstract

Multipath TCP [RFC6824] defines the ADD_ADDR option that allows a host to announce its addresses to the remote host. In this document we propose some improvements to this mechanism.

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 http://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 January 9, 2017.

Copyright Notice

Table of Contents

1. Introduction

Multipath TCP is an extension to TCP [RFC0793] that was specified in [RFC6824]. Multipath TCP allows hosts to use multiple paths to send and receive the data belonging to one connection. For this, a Multipath TCP is composed of several TCP connections that are called subflows. [RFC6824] defines two options to manage the host addresses:

- ADD_ADDR is used to announce one address bound to a host (possibly combined with a port number)
- REMOVE_ADDR is used to indicate that an address previously attached to a host is not anymore attached to this host

To cope with Network Address Translation (NAT), the ADD_ADDR and REMOVE_ADDR options contain an address identifier encoded as an 8 bits integer.

When the initial subflow is created, it is assumed to be initiated from the address of the client whose identifier is 0 towards the address of the server whose identifier is also 0. Both the client and the server can use ADD_ADDR to advertise the other addresses that they use. When an additional subflow is created, the MP_JOIN option placed in the SYN (resp. SYN+ACK) contains the identifier of the address used to create (resp. accept) the subflow.

The latest Multipath TCP draft [I-D.ietf-mptcp-rfc6824bis] defines the ADD_ADDR option as shown in Figure 1.

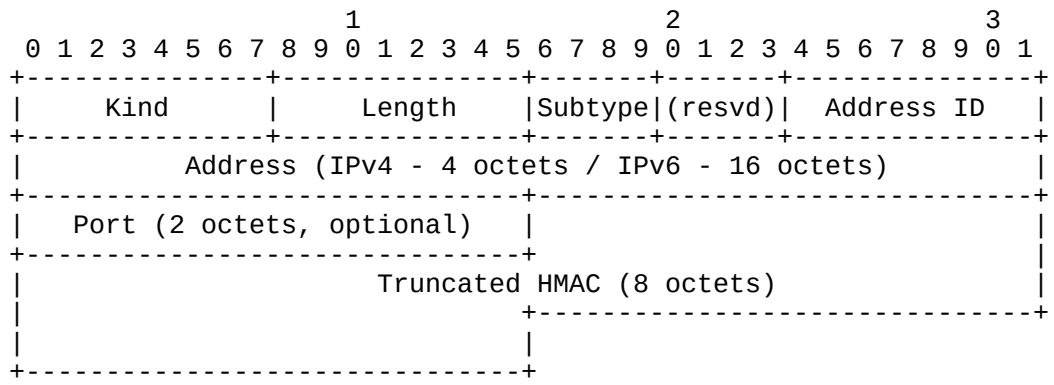

Figure 1: The Multipath TCP ADD_ADDR option format

In this document, we propose to slightly modify the format of this option based on issues that have been detected while working with the Multipath TCP implementation in the Linux kernel. More precisely, we address four different problems. The first, discussed in Section 2.1, is that the ADD_ADDR option is sent unreliably. This implies that the host sending an ADD_ADDR option cannot be sure that the address that it has advertised has been learned by the distant host. The second issue, discussed in Section 2.2 is the handling of backup subflows. Multipath TCP supports the creation of backup subflows through the B bit in the MP_JOIN option. These backup subflows consume energy and radio ressources on mobile devices and it would be useful for a host to be able to advertise a backup address that would be used to create subflows after a failure. The third issue is that multihomed hosts may have preferences on the utilisation of some of their addresses/interfaces to create additional subflows. Section 2.3 proposes a priority field that allows them to advertise these preferences. The fourth issue is that multihomed hosts, especially with IPv6, often have several addresses assigned to each interface. In this case, it can be difficult to establish disjoint paths between the communicating hosts. Section 2.4 proposes a community field in the ADD_ADDR option to indicate that some addresses share the same path. The last issue, discussed in Section 2.5 is that some hosts, e.g. servers behind a load balancer or clients behind a firewall, may want to indicate that the address used for the initial subflow should not be used to create additional ones.

2. Proposed ADD_ADDR option

To cope with the issues described later in this document we propose a new format for this option. The format for this new option is shown in Figure 2.

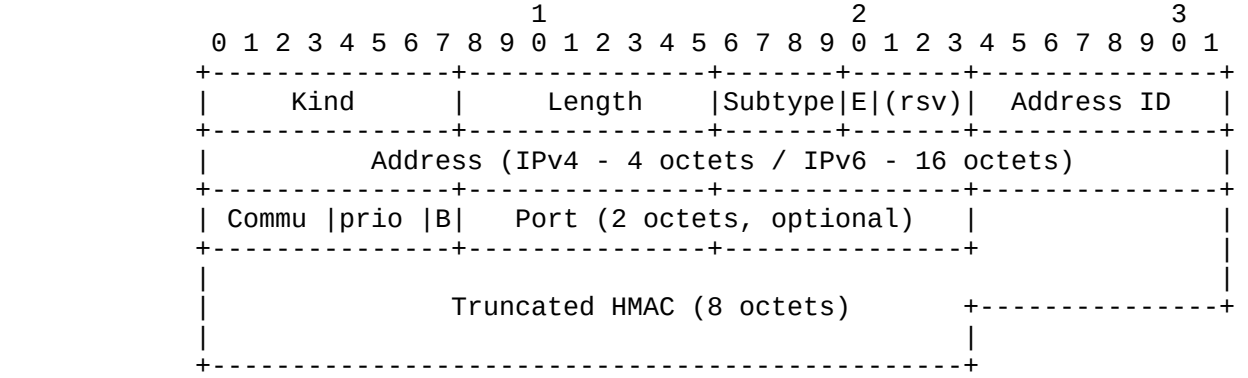

Figure 2: The proposed Multipath TCP new ADD_ADDR option format

2.1. Reliability

A first issue with the ADD_ADDR option is that since it is transmitted as a TCP option, it is not delivered reliably [Cellnet12]. When a host announces an IPv4 address, it can insert the ADD_ADDR option inside a segment that carries data that would thus be delivered reliably like user data. However, if the ADD_ADDR option contains an IPv6 address, it might be too large to fit inside a segment that already contains a DSS option and possibly other options such as the [RFC1323] timestamps. Given its length, the ADD_ADDR option cannot be placed in the same segment as a DSS option. In these two cases, the ADD_ADDR option will be often transmitted inside a duplicate ACK that is not delivered reliably. [Cellnet12] proposes a method to improve the reliability of the transmission of the ADD_ADDR option, but to our knowledge this method has never been implemented. To cope with packet losses, we propose to rely on the "E" (Echo) flag in the ADD_ADDR option (Figure 3).

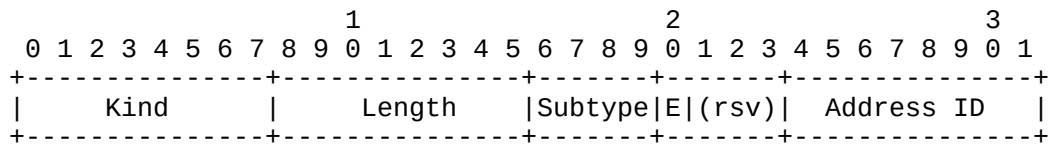

Figure 3: The part of the proposed Multipath TCP new ADD_ADDR option format with the Echo flag

The "E" flag is the "Echo" flag. When set to 0, it indicates that the host sending this option is advertising a new address to the receiving host. When set to 1, it indicates that the host sending this option acknowledges the reception of an ADD_ADDR option by echoing it. Upon reception of an ADD_ADDR option without the "E" flag set, the receiving host MUST return the exact option that it received with the "E" flag set to 1 to indicate the reception of the ADD_ADDR option. If an host advertising a new address does nt receive an echo, or receives an invalid echo of the option it MAY retransmit the ADD_ADDR. To cope with the loss of the echo of the option, if an host that advertised a new address without receiving the echo receives an MP_JOIN on this address, it MUST consider this address as having been echoed, and MUST NOT retransmit this ADD_ADDR again.

2.2. Backup

The subflows that compose a Multipath TCP connection are not all equal. [RFC6824] defines two types of subflows:

- the regular subflows
- the backup subflows

The regular subflows can be used to transport any data. The backup subflows are intended to be used only when all the regular subflows fail. [RFC6824] defines them by using the following sentence: "Hosts can indicate at initial subflow setup whether they wish the subflow to be used as a regular or backup path - a backup path only being used if there are no regular paths available."

In [RFC6824] a host can specify the type of a subflow during the three-way-handshake by using the "B" flag of the MP_JOIN option as shown in Figure 4 and Figure 5 or when the subflow is already established by sending an MP_PRIO option shown in Figure 6.

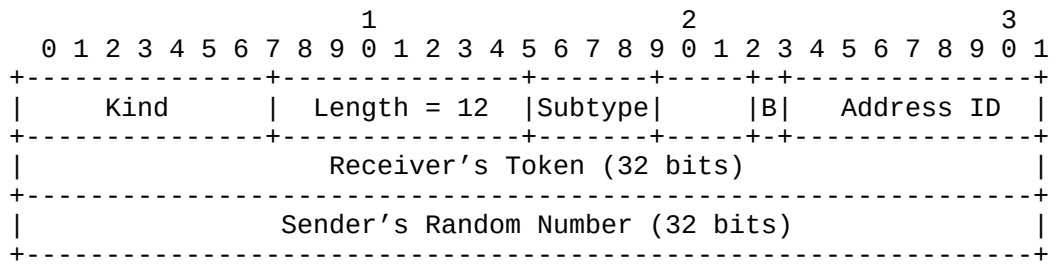

Figure 4: Join Connection (MP_JOIN) Option (for Initial SYN)

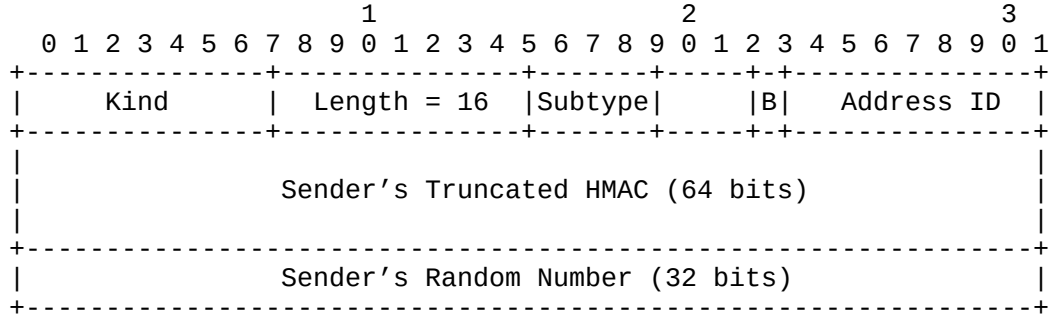

Figure 5: Join Connection (MP_JOIN) Option (for Responding SYN/ACK)

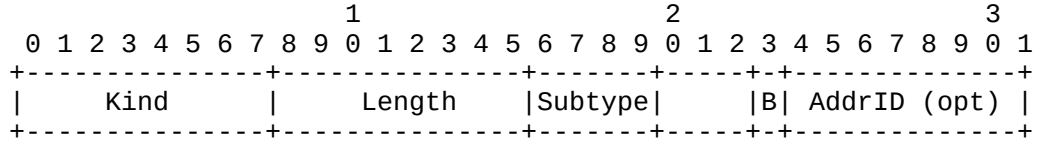

Figure 6: Change Subflow Priority (MP_PRIO) Option

Both solutions rely on the principle that a subflow can be set in backup mode only when being already established or during the subflow setup. On mobile devices, backup subflows consume radio ressources when they are established. This could unnecessarily consume both energy on the mobile device [ATC14] and radio ressources in the network for subflows that do not carry any data. Measurements on smartphones [PAM2016] indicate that many subflows do not carry any data but still consume resources for the SYN, RST and FIN packets.

To allow hosts using Multipath TCP to save ressources, we propose to add the "B" "Backup" Flag in the ADD_ADDR option as shown in Figure 7. This would allow an host to save ressources by being aware

of the remote backup addresses that could be used if all the non-backup subflows fail without having to establish a subflow, achieving a break-before-make scheme.

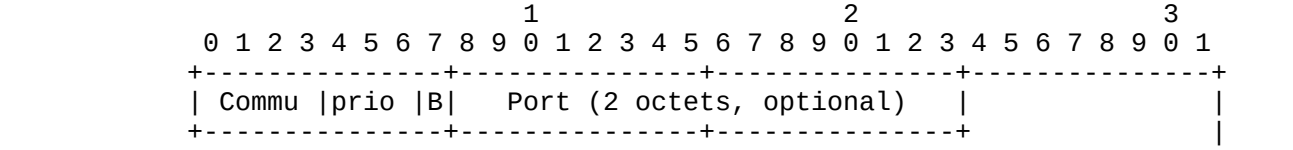

Figure 7: The part of the proposed Multipath TCP new ADD_ADDR option format with the Backup flag

2.3. Priorities

The backup mode defined in [RFC6824] only supports an "all-or-nothing" mode in the usage of the subflows, where an host might just prefer to use certain subflow over others.

To allow an host to inform the receving host about its preference in terms of subflow usage, we propose to modify the ADD_ADDR option by adding 3 "priority" bits as shown in Figure 8.

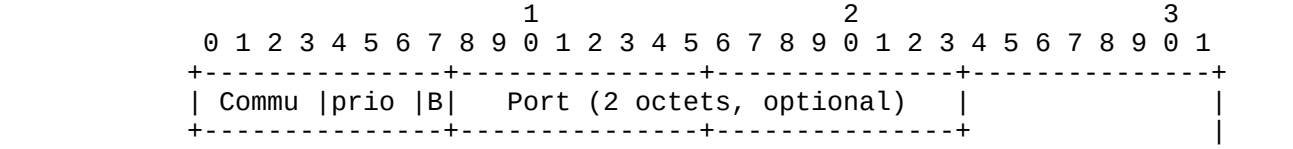

Figure 8: The part of the proposed Multipath TCP new ADD_ADDR option format with the priority bits

This host MAY use this priority to determine when to establish a subflow towards this address. The priority field MUST be interpreted as an unsigned integer value with the highest numerical value being the most preferred one.

To allow the priority of an already established subflow to be modified, we propose to modify the MP_PRIO option by adding the 3 priority bits next to the "B" flag has shown in Figure 9.

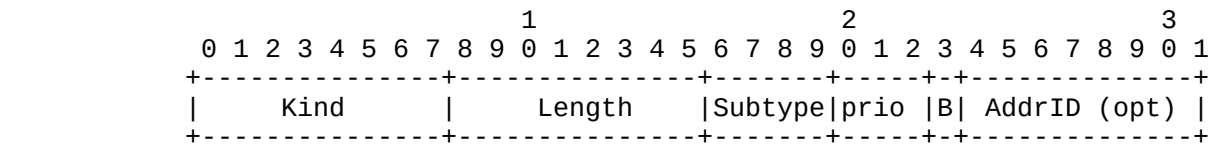

Figure 9: Change Subflow Priority (MP_PRIO) Option with 3 priority bits added

To allow the hosts to advertise a per-subflow priority during the three-way-handshake we modify the MP_JOIN option by adding the 3 priority bits as shown in Figure 10 and Figure 11,

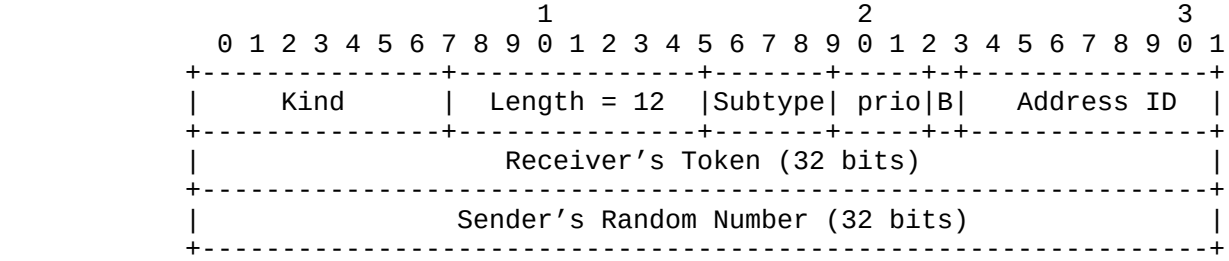

Figure 10: Join Connection (MP_JOIN) Option (for Initial SYN) with the 3 priority bits

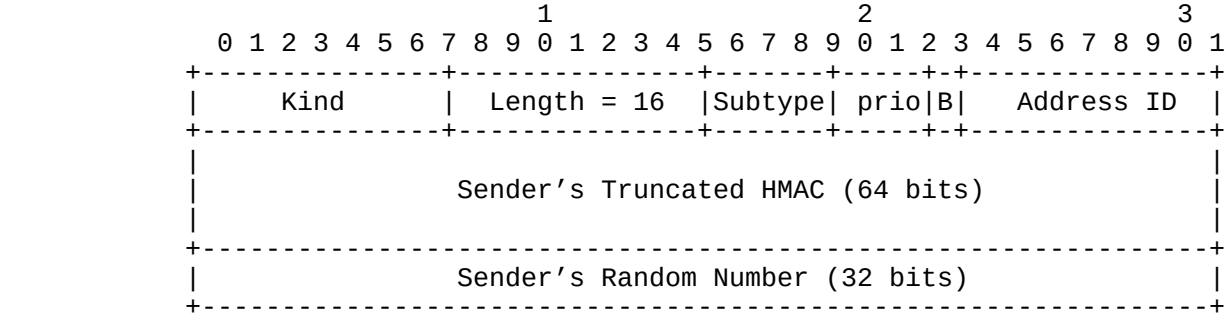

Figure 11: Join Connection (MP_JOIN) Option (for Responding SYN/ACK) with the 3 priority bits

The priority bits included in the MP_JOIN specify indicate the priority associated to this subflow. A host MAY use this information when scheduling packets over this particular subflow.

2.4. Path diversity

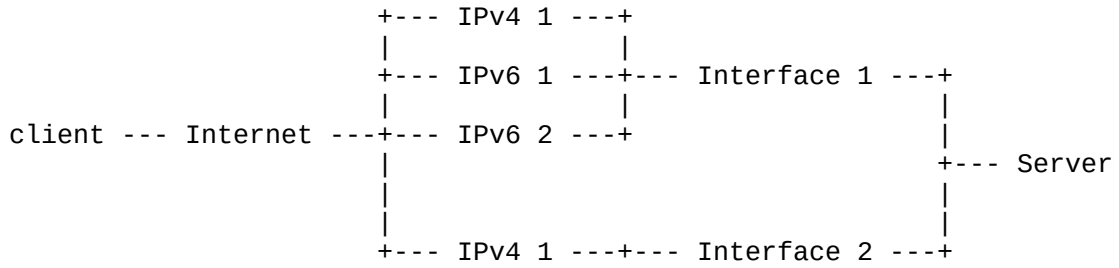

Figure 12: A dual stack server with multiple IP addresses attached to the same interface.

As shown in Figure 12 a host might have several IP addresses assigned to a single interface. Some clients would like to be able to create subflows over disjoint paths to maximise the diversity of the subflows. With the current ADD_ADDR option, the host receiving several ADD_ADDR has no way of knowing the diversity between these path. In the case of Figure 12 it could end up establishing 4 subflows where 2 could be sufficient to maximise diversity.

To allow a host to inform the receiving host about the diversity of several addresses we propose to modify the ADD_ADDR to include 4 bits describing a "Community" associated to this address. The community values are an opaque field and it is expected that two addresses having the same community share some resources.

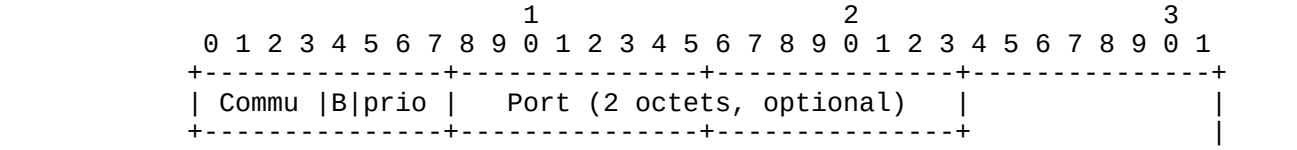

Figure 13: The part of the proposed Multipath TCP new ADD_ADDR option format with the Community bits

With the community bits, a dual-stack host could elect to regroup all the addresses attached to a single interface under the same community, allowing the receiving host to decide on which advertised addresses it wants to establish new subflows.

2.5. Load balancing

Many large web sites are served by servers that are behind a load balancer. The load balancer receives the connection establishment attempts and forwards them to the actual servers that serve the requests. One issue for the end-to-end deployment of Multipath TCP is it ability to be used on load-balancers. Different types of load balancers are possible. We consider a simple but important load balancer than does not maintain any per-flow state. This load balancer is illustrated in illustrated in Figure 14. A stateless load balancer can be implemented by hashing the five tuple (IP addresses and port numbers) of each incoming packet and forward them to one of the servers based on the hash value computed. With TCP, this load balancer ensures that all the packets that belong to one TCP connection are sent to the same server.

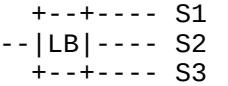

Figure 14: Stateless load balancer

With Multipath TCP, this approach cannot be used anymore when subflows are created by the clients. Such subflows can use any five tuple and thus packets belonging to them will be forwarded over any server, not necessarily the one that was selected by the hashing function for the initial subflow.

To allow Multipath TCP to work for hosts being hosted behind unmodified layer 4 load balancers, we propose to use the unused "B" flag in the MP_CAPABLE option sent (shown in Figure 15 in the SYN+ACK. This flag would allow a host behind a layer 4 load balancer to inform the other host that this address MUST NOT be used to create additional subflows.

A host receiving an MP_CAPABLE with the "B" set to 1 MUST NOT try to establish a subflow to the address used in the MP_CAPABLE. This bit can also be used in the MP_CAPABLE option sent in the SYN by a client that resides behind a NAT or firewall or does not accept server-initiated subflows.

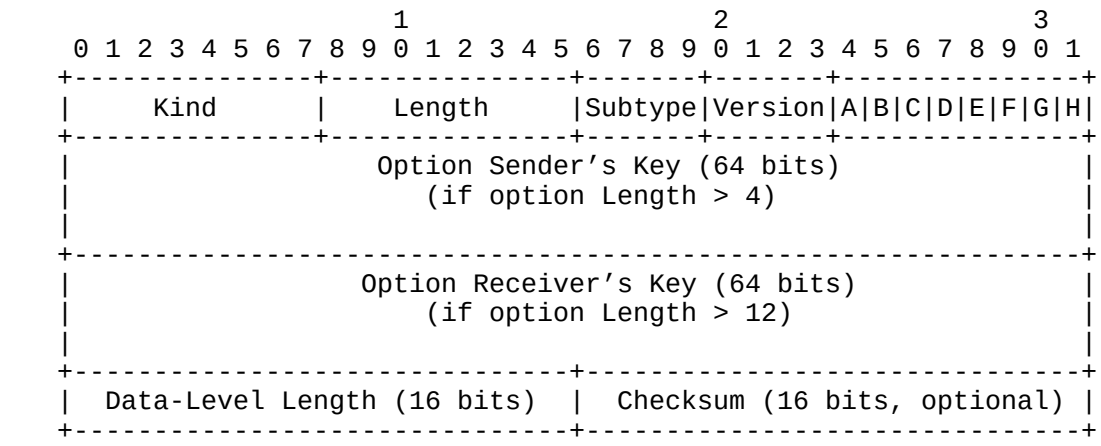

Figure 15: Multipath Capable (MP_CAPABLE) Option

This bit can be used by the servers behind a stateless load balancers. Each of these servers has a different IP address than the address of the load balancer. The servers set the "B" flag in the MP_CAPABLE option that they return and advertise their own address by using the ADD_ADDR option. Upon reception of this option, the clients can create the additional subflows towards these addresses. Compared with current stateless load balancers, an advantage of this approach is that the packets belonging to the additional subflows do not need to pass through the load balancer.

3. IANA considerations

This document proposes some modifications to the Multipath TCP options defined in [RFC6824]. These modifications do not require any specific action from IANA.

4. Security considerations

The security considerations defined for Multipath TCP in [RFC6182] and [RFC7430] are applicable.

The "E" flag, community and priority values in the ADD_ADDR option do not change the security considerations for the handling of this option. Since the ADD_ADDR option is protected by an HMAC, an off-path attacker cannot inject such an option in an existing Multipath TCP connection.

The "priority" field of the MP_PRIO option is not protected by a HMAC. It could be useful to consider the utilisation of an HMAC to protect this option like the ADD_ADDR option.

The "B" flag of the MP_CAPABLE option does not change the security considerations of this option. If an attacker that resides on a path sets this bit, it could prevent the establishment of subflows. However, Multipath TCP does not protect against an attacker that resides on the path of the initial subflow and can modify the SYN/ SYN+ACK packets.

5. Conclusion

In this document, we have discussed several issues with the advertisement of addresses with the address advertisement in Multipath TCP. We have proposed several modifications to the protocol to address these issues.

6. References

6.1. Normative References

[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>.

6.2. Informative References

[ATC14] Yeon-sup Lim, ., Yung-Chih Chen, ., Nahum, Erich., Don Towsley, ., and . Richard Gibbens, "How green is multipath TCP for mobile devices?", AllThingsCellular14 , 2014.

[Cellnet12] Paasch, C., Detal, G., Duchene, F., Raiciu, C., and O. Bonaventure, "Exploring Mobile/WiFi Handover with Multipath TCP", ACM SIGCOMM workshop on Cellular Networks (Cellnet12) , 2012, <http://inl.info.ucl.ac.be/publications/exploring-mobilewifi-handover-multipath-tcp>.

[I-D.ietf-mptcp-rfc6824bis] Ford, A., Raiciu, C., Handley, M., Bonaventure, O., and C. Paasch, "TCP Extensions for Multipath Operation with Multiple Addresses", draft-ietf-mptcp-rfc6824bis-06 (work in progress), July 2016.

[PAM2016] Quentin De Coninck, ., Matthieu Baerts, ., Benjamin Hesmans, ., and O. Bonaventure, "A First Analysis of Multipath TCP on Smartphones", 17th International Passive and Active Measurements Conference , April 2016, <http://inl.info.ucl.ac.be/publications/first-analysis-multipath-tcp-smartphones>.

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

[RFC1323] Jacobson, V., Braden, R., and D. Borman, "TCP Extensions for High Performance", RFC 1323, DOI 10.17487/RFC1323, May 1992, <http://www.rfc-editor.org/info/rfc1323>.

[RFC6182] Ford, A., Raiciu, C., Handley, M., Barre, S., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development", RFC 6182, DOI 10.17487/RFC6182, March 2011, <http://www.rfc-editor.org/info/rfc6182>.

[RFC7430] Bagnulo, M., Paasch, C., Gont, F., Bonaventure, O., and C. Raiciu, "Analysis of Residual Threats and Possible Fixes for Multipath TCP (MPTCP)", RFC 7430, DOI 10.17487/RFC7430, July 2015, <http://www.rfc-editor.org/info/rfc7430>.

Authors' Addresses

Fabien Duchene
UCLouvain

Email: fabien.duchene@uclouvain.be

Olivier Bonaventure
UCLouvain

Email: Olivier.Bonaventure@uclouvain.be