Network Working Group	M. Bagnulo
Internet-Draft	UC3M
Intended status: Informational	January 26, 2011
Expires: July 30, 2011	

TOC

Threat Analysis for TCP Extensions for Multi-path Operation with Multiple Addresses draft-ietf-mptcp-threat-08

Abstract

Multi-path TCP (MPTCP for short) describes the extensions proposed for TCP so that endpoints of a given TCP connection can use multiple paths to exchange data. Such extensions enable the exchange of segments using different source-destination address pairs, resulting in the capability of using multiple paths in a significant number of scenarios. Some level of multihoming and mobility support can be achieved through these extensions. However, the support for multiple IP addresses per endpoint may have implications on the security of the resulting MPTCP protocol. This note includes a threat analysis for MPTCP.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

TOC

Multi-path TCP (MPTCP for short) describes the extensions proposed for TCP [RFC0793] (Postel, J., “Transmission Control Protocol,” September 1981.) so that endpoints of a given TCP connection can use multiple paths to exchange data. Such extensions enable the exchange of segments using different source-destination address pairs, resulting in the capability of using multiple paths in a significant number of scenarios. Some level of multihoming and mobility support can be achieved through these extensions. However, the support for multiple IP addresses per endpoint may have implications on the security of the resulting MPTCP protocol. This note includes a threat analysis for MPTCP. There are there may other ways to provide multiple paths for a TCP connection other than the usage of multiple addresses. The threat analysis performed in this document is limited to the specific case of using multiple addresses per endpoint.

2. Scope

There are multiple ways to achieve Multi-path TCP. Essentially what is needed is for different segments of the communication to be forwarded through different paths by enabling the sender to specify some form of path selector. There are multiple options for such a path selector, including the usage of different next hops, using tunnels to different egress points and so on. In this note, we will focus on a particular approach, namely MPTCP, that rely on the usage of multiple IP address per endpoint and that uses different source-destination address pairs as a mean to express different paths. So, in the rest of this note, the MPTCP expression will refer to this Multi-addressed flavour of Multi-path TCP [I-D.ietf-mptcp-multiaddressed] (Ford, A., Raiciu, C., and M. Handley, "TCP Extensions for Multipath Operation with Multiple Addresses," October 2010.).

In this note we perform a threat analysis for MPTCP. Introducing the support of multiple addresses per endpoint in a single TCP connection may result in additional vulnerabilities compared to single-path TCP. The scope of this note is to identify and characterize these new vulnerabilities. So, the scope of the analysis is limited to the additional vulnerabilities resulting from the multi-address support compared to the current TCP protocol (where each endpoint only has one address available for use per connection). A full analysis of the complete set of threats is explicitly out of the scope. The goal of this analysis is to help the MPTCP protocol designers create an MPTCP specification that is as secure as the current TCP. It is a non-goal of this analysis to help in the design of MPTCP that is more secure than regular TCP.

We will focus on attackers that are not along the path, at least not during the whole duration of the connection. In the current single path TCP, an on-path attacker can launch a significant number of attacks, including eavesdropping, connection hijacking Man-in-the-Middle attacks and so on. However, it is not possible for the off-path attackers to launch such attacks. There is a middle ground in case the attacker is located along the path for a short period of time to launch the attack and then moves away, but the attack effects still apply. These are the so-called time-shifted attacks. Since these are not possible in today's TCP, we will also consider them as part of the analysis. So, summarizing, we will consider both attacks launched by off-path attackers and time-shifted attacks. Attacks launched by on-path attackers are out of scope, since they also apply to current single-path TCP.

However, that some current on-path attacks may become more difficult with multi-path TCP, since an attacker (on a single path) will not have visibility of the complete data stream.

3. Related work

There is a significant amount of previous work in terms of analysis of protocols that support address agility. In this section we present the most relevant ones and we relate them to the current MPTCP effort. Most of the problems related to address agility have been deeply analyzed and understood in the context of Route Optimization support in Mobile IPv6 (MIPv6 RO) [RFC3775] (Johnson, D., Perkins, C., and J. Arkko, “Mobility Support in IPv6,” June 2004.). [RFC4225] (Nikander, P., Arkko, J., Aura, T., Montenegro, G., and E. Nordmark, “Mobile IP Version 6 Route Optimization Security Design Background,” December 2005.) includes the rational for the design of the security of MIPv6 RO. All the attacks described in the aforementioned analysis apply here and are an excellent basis for our own analysis. The main differences are:

*In MIPv6 RO, the address binding affects all the communications involving an address, while in the MPTCP case, a single connection is at stake. If a binding between two addresses is created at the IP layer, this binding can and will affect all the connections that involve those addresses. However, in MPTCP, if an additional address is added to an ongoing TCP connection, the additional address will/can only affect the connection at hand and not other connections even if the same address is being used for those other connections. The result is that in MPTCP there is much less at stake and the resulting vulnerabilities are less. On the other hand, it is very important to keep the assumption valid that the address bindings for a given connection do not affect other connections. If reusing of binding or security information is to be considered, this assumption could be no longer valid and the full impact of the vulnerabilities must be assessed.

*In MIPv6 there is a trusted third party, called the Home Agent that can help with some security problems, as expanded in the next bullet.

*In MIPv6 RO, there is the assumption that the original address (Home Address) through which the connection has been established is always available and in case it is not, the communication will be lost. This is achieved by leveraging in the on the trusted party (the Home Agent) to rely the packets to the current location of the Mobile Node. In MPTCP, it is an explicit goal to provide communication resilience when one of the address pairs is no longer usable, so it is not possible to leverage on the original address pair to be always working.

*MIPv6 RO is of course designed for IPv6 and it is an explicit goal of MPTCP to support both IPv6 and IPv4. Some MIPv6 RO security solutions rely on the usage of some characteristics of IPv6 (such as the usage of CGAs [RFC3972] (Aura, T., “Cryptographically Generated Addresses (CGA),” March 2005.)), which will not be usable in the context of MPTCP.

* As opposed to MPTCP, MIPv6 RO does not have a connection state information, including sequence numbers, port numbers that could be leveraged to provide security in some form.

In the Shim6 [RFC5533] (Nordmark, E. and M. Bagnulo, “Shim6: Level 3 Multihoming Shim Protocol for IPv6,” June 2009.) design, similar issues related to address agility were considered and a threat analysis was also performed [RFC4218] (Nordmark, E. and T. Li, “Threats Relating to IPv6 Multihoming Solutions,” October 2005.). The analysis performed for Shim6 also largely applies to the MPTCP context, the main difference being:

* The Shim6 protocol is a layer 3 protocol so all the communications involving the target address are at stake; in MPTCP, the impact can be limited to a single TCP connection.

* Similarly to MIPv6 RO, Shim6 only uses IPv6 addresses as identifiers and leverages on some of their properties to provide the security, such as relying on CGAs or HBAs [RFC5535] (Bagnulo, M., “Hash-Based Addresses (HBA),” June 2009.), which is not possible in the MPTCP case where IPv4 addresses must be supported.

* Similarly to MIPv6 RO, Shim6 does not have a connection state information, including sequence numbers, port that could be leveraged to provide security in some form.

SCTP [RFC4960] (Stewart, R., “Stream Control Transmission Protocol,” September 2007.)is a transport protocol that supports multiple addresses per endpoint and the security implications are very close to the ones of MPTCP. A security analysis, identifying a set of attacks and proposed solutions was performed in [RFC5062] (Stewart, R., Tuexen, M., and G. Camarillo, “Security Attacks Found Against the Stream Control Transmission Protocol (SCTP) and Current Countermeasures,” September 2007.). The results of this analysis apply directly to the case of MPTCP. However, the analysis was performed after the base SCTP protocol was designed and the goal of the document was essentially to improve the security of SCTP. As such, the document is very specific to the actual SCTP specification and relies on the SCTP messages and behaviour to characterize the issues. While some them can be translated to the MPTCP case, some may be caused by specific behaviour of SCTP. So, the conclusion is that while we do have a significant amount of previous work that is closely related and we can and will use it as a basis for this analysis, there is a set of characteristics that are specific to MPTCP that grant the need for a specific analysis for MPTCP. The goal of this analysis is to help MPTCP protocol designers to include a set of security mechanisms that prevent the introduction of new vulnerabilities to the Internet due to the adoption of MPTCP.

4. Basic MPTCP.

TOC

The goal of this document is to serve as input for MPTCP protocol designers to properly take into account the security issues. As such, the analysis cannot be performed for a specific MPTCP specification, but must be a general analysis that applies to the widest possible set of MPTCP designs. We will characterize what are the fundamental features that any MPTCP protocol must provide and attempt to perform the security implications only assuming those. In some cases, we will have a design choice that will significantly influence the security aspects of the resulting protocol. In that case we will consider both options and try to characterize both designs.
We assume that any MPTCP will behave in the case of a single address per endpoint as TCP. This means that a MPTCP connection will be established by using the TCP 3-way handshake and will use a single address pair.
The addresses used for the establishment of the connection do have a special role in the sense that this is the address used as identifier by the upper layers. The address used as destination address in the SYN packet is the address that the application is using to identify the peer and has been obtained either through the DNS (with or without DNSSEC validation) or passed by a referral or manually introduced by the user. As such, the initiator does have a certain amount of trust in the fact that it is establishing a communication with that particular address. If due to MPTCP, packets end up being delivered to an alternative address, the trust that the initiator has placed on that address would be deceived. In any case, the adoption of MPTCP necessitates a slight evolution of the traditional TCP trust model, in that the initiator is additionally trusting the peer to provide additional addresses which it will trust to the same degree as the original pair. An application or implementation that cannot trust the peer in this way should not make use of multiple paths.
During the 3-way handshake, the sequence number will be synchronized for both ends, as in regular TCP. We assume that a MPTCP connection will use a single sequence number for the data, even if the data is exchanged through different paths, as MPTCP provides an in-order delivery service of bytes
Once the connection is established, the MPTCP extensions can be used to add addresses for each of the endpoints. This is achieved by each end sending a control message containing the additional address(es). In order to associate the additional address to an ongoing connection, the

connection needs to be identified. We assume that the connection can be identified by the 4-tuple of source address, source port, destination address, destination port used for the establishment of the connection. So, at least, the control message that will convey the additional address information can also contain the 4-tuple in order to inform about what connection the address belong to (if no other connection identifier is defined). There are two different ways to convey address information:

- Explicit mode: the control message contain a list of addresses.
- Implicit mode: the address added is the one included in the source address field of the IP header

These two modes have different security properties for some type of attacks. The explicit mode seems to be the more vulnerable to abuse. The implicit mode may benefit from forms of ingress filtering security, which would reduce the possibility of an attacker to add any arbitrary address to an ongoing connection. However, ingress filtering deployment is far from universal, and it is unwise to rely on it as a basis for the protection of MPTCP.
Further consideration about the interaction between ingress filtering and implicit mode signaling is needed in the case that we need to remove an address that is no longer available from the MPTCP connection. A host attached to a network that performs ingress filtering and using implicit signaling would not be able to remove an address that is no longer available (either because of a failure or due to a mobility event) from an ongoing MPTCP connection.
We will assume that MPTCP will use all the address pairs that it has available for sending packets and that it will distribute the load based on congestion among the different paths.

---

5. Flooding attacks

TOC

The first type of attacks that are introduced by address agility are the flooding (or bombing) attacks. The setup for this attack is depicted in the following figure:

```
             +--------+        (step 1)          +------+
             |Attacker| ------------------------- |Source|
             |    A   |IPA                    IPS|  S   |
             +--------+                          /+------+
                                                /
                                  (step 2) /
                                              /
                                             v IPT
                                         +------+
                                         |Target|
                                         |  T   |
                                         +------+
```

The scenario consists of an attacker A who has an IP address IPA. A server that can generate a significant amount of traffic (such as a streaming server), called source S and that has IP address IPS. Target T has an IP address IPT.
In step 1 of this attack, the attacker A establishes a MPTCP connection with the source of the traffic server S and starts downloading a significant amount of traffic. The initial connection only involves one IP address per endpoint, IPA and IPS. Once that the download is on course, in the step 2 of the attack is that the attacker A adds IPT as one of the available addresses for the communication. How the additional address is added depends on the MPTCP address management mode. In explicit address management, the attacker A only needs to send a signaling packet conveying address IPT. In implicit mode, the attacker A would need to send a packet with IPT as the source address. Depending on whether ingress filtering is deployed and the location of the attacker, it may be possible or not for the attacker to send such a packet. At this stage, the MPTCP connection still has a single address for the Source S i.e. IPS but has two addresses for the Attacker A, IPA and IPT. The attacker now attempts to get the Source S to send the traffic of the ongoing download to the Target T IP address i.e. IPT. The attacker can do that by pretending that the path between IPA and IPT is congested but that the path between IPS and IPT is not. In order to do that, it needs to send ACKs for the data that flows through the path between IPS and IPT and do not send ACKs for the data that is sent to IPA. The details of this will depend on how the data sent through the different paths is ACKed. One possibility is that ACKs for the data sent using a given address pair should come in packets containing the same address pair. If so, the attacker would need to send ACKs using packets containing IPT as the source address to keep the attack flowing. This may be possible or not depending on the deployment of ingress filtering and the location of the attacker. The attacker would also need to guess the sequence number of the data being sent to the Target. Once the attacker manages to perform these actions the attack

is on place and the download will hit the target. In this type of attacks, the Source S still thinks it is sending packets to the Attacker A while in reality it is sending the packet to Target T.
Once that the traffic from the Source S start hitting the Target T, the target will react. Since the packets are likely to belong to a non existent TCP connection, the Target T will issue RST packets. It is relevant then to understand how MPTCP reacts to incoming RST packets.
It seems that the at least the MPTCP that receives a RST packet should terminate the packet exchange corresponding to the particular address pair (maybe not the complete MPTCP connection, but at least it should not send more packets with the address pair involved in the RST packet). However, if the attacker, before redirecting the traffic has managed to increase the window size considerably, the flight size could be enough to impose a significant amount of traffic to the Target node.
There is a subtle operation that the attacker needs to achieve in order to launch a significant attack. On the one hand it needs to grow the window enough so that the flight size is big enough to cause enough effect and on the other hand the attacker needs to be able to simulate congestion on the IPA-IPS path so that traffic is actually redirected to the alternative path without significantly reducing the window. This will heavily depend on how the coupling of the windows between the different paths works, in particular how the windows are increased.
Some designs of the congestion control window coupling could render this attack ineffective. If the MPTCP protocol requires performing slow start per subflow, then the flooding will be limited by the slow-start initial window size.
Previous protocols, such as MIPv6 RO and SCTP, that have to deal with this type of attacks have done so by adding a reachability check before actually sending data to a new address. The solution used in other protocols, would include the Source S to explicitly asking the host sitting in the new address (the Target T sitting in IPT) whether it is willing to accept packets from the MPTCP connection identified by the 4-tuple IPA, port A, IPS, port S. Since this is not part of the established connection that Target T has, T would not accept the request and Source S would not use IPT to send packets for this MPTCP connection. Usually, the request also includes a nonce that cannot be guessed by the attacker A so that it cannot fake the reply to the request easily. In the case of SCTP, it sends a message with a 64-bit nonce (in a HEARTBEAT).
One possible approach to do this reachability test would be to perform a 3-way handshake for each new address pair that is going to be used in a MPTCP connection. While there are other reasons for doing this (such as NAT traversal), such approach would also act as a reachability test and would prevent the flooding attacks described in this section.
Another type of flooding attack that could potentially be performed with MPTCP is one where the attacker initiates a communication with a peer and includes a long list of alternative addresses in explicit mode. If the peer decides to establish subflows with all the available addresses, the attacker have managed to achieve an amplified attack,

since by sending a single packet containing all the alternative addresses it triggers the peer to generate packets to all the destinations.

6. Hijacking attacks

TOC

6.1. Hijacking attacks to the Basic MPTCP protocol

TOC

The hijacking attacks essentially use the MPTCP address agility to allow an attacker to hijack a connection. This means that the victim of a connection thinks that it is talking to a peer, while it is actually exchanging packets with the attacker. In some sense it is the dual of the flooding attacks (where the victim thinks it is exchanging packets with the attacker but in reality is sending the packets to the target). The scenario for a hijacking attack is described in the next figure.

```
              +-------+                          +-------+
              | Node  | ------------------------ | Node  |
              |   1   |IP1                    IP2|   2   |
              +-------+                         /+-------+
                                               /
                                              /
                                             /
                                            v IPA
                                       +---------+
                                       |Attacker |
                                       |    A    |
                                       +---------+
```

We have a MPTCP connection established between Node 1 and Node 2. The connection is using only one address per endpoint, IP1 and IP2. The attacker then launches the hijacking attack by adding IPA as an additional address for Node 1. There is not much difference between explicit or implicit address management, since in both cases the Attacker A could easily send a control packet adding the address IPA, either as control data or as the source address of the control packet. In order to be able to hijack the connection, the attacker needs to know the 4-tuple that identifies the connection, including the pair of addresses and the pair of ports. It seems reasonable to assume that knowing the source and destination IP addresses and the port of the server side is fairly easy for the attacker. Learning the port of the client (i.e. of the initiator of the connection) may prove to be more challenging. The attacker would need to guess what the port is or to learn it by intercepting the packets. Assuming that the attacker can gather the 4-tuple and issue the message adding IPA to the addresses available for the MPTCP connection, then the attacker A has been able to participate in the communication. In particular:

- *Segments flowing from the Node 2:Depending how the usage of addresses is defined, Node 2 will start using IPA to send data to. In general, since the main goal is to achieve multi-path capabilities, we can assume that unless there are already many IP address pairs in use in the MPTCP connection, Node 2 will start sending data to IPA. This means that part of the data of the communication will reach the Attacker but probably not all of it. This already has negative effects, since Node 1 will not receive all the data from Node 2. Moreover, from the application perspective, this would result in DoS attack, since the byte flow will stop waiting for the missing data. However, it is not enough to achieve full hijacking of the connection, since part of data will be still delivered to IP1, so it would reach Node 1 and not the Attacker. In order for the attacker to receive all the data of the MPTCP connection, the Attacker must somehow remove IP1 of the set of available addresses for the connection. in the case of implicit address management, this operation is likely to imply sending a termination packet with IP1 as source address, which may or not be possible for the attacker depending on whether ingress filtering is in place and the location of the attacker. If explicit address management is used, then the attacker will send a remove address control packet containing IP1. Once IP1 is removed, all the data sent by Node 2 will reach the Attacker and the incoming traffic has been hijacked.

- *Segments flowing to the Node 2: As soon as IPA is accepted by Node 2 as part of the address set for the MPTCP connection, the Attacker can send packets using IPA and those packets will be considered by Node 2 as part of MPTCP connection. This means that the attacker will be able to inject data into the MPTCP connection, so from this perspective, the attacker has hijacked part of the outgoing traffic. However, Node 1 would still be able to send traffic that will be received by Node 2 as part of the MPTCP connection. This means that there will be two sources of data i.e. Node 1 and the attacker, potentially preventing the full hijacking of the outgoing traffic by the attacker. In order to achieve a full hijacking, the attacker would need to remove IP1 from the set of available addresses. This can be done using the same techniques described in the previous paragraph.

A related attack that can be achieved using similar techniques would be a Man-in-the-Middle (MitM) attack. The scenario for the attack is depicted in the figure below.

```
+------+                 +------+
| Node | --------------- | Node |
|   1  |IP1           IP2|  2   |
+------+ \              /+------+
          \            /
           \          /
            \        /
            v IPA  v
           +--------+
           |Attacker|
           |    A   |
           +--------+
```

There is an established connection between Node 1 and Node 2. The Attacker A will use the MPTCP address agility capabilities to place itself as a MitM. In order to do so, it will add IP address IPA as an additional address for the MPTCP connection on both Node 1 and Node 2. This is essentially the same technique described earlier in this section, only that it is used against both nodes involved in the communication. The main difference is that in this case, the attacker can simply sniff the content of the communication that is forwarded through it and in turn forward the data to the peer of the communication. The result is that the attacker can place himself in the middle of the communication and sniff part of the traffic unnoticed. Similar considerations about how the attacker can manage to get to see all the traffic by removing the genuine address of the peer apply.

6.2. Time-shifted hijacking attacks

TOC

A simple way to prevent off-path attackers to launch hijacking attacks is to provide security of the control messages that add and remove addresses by the usage of a cookie. In this type of approaches, the peers involved in the MPTCP connection agree on a cookie, that is exchanged in plain text during the establishment of the connection and that needs to be presented in every control packet that adds or removes an address for any of the peers. The result is that the attacker needs to know the cookie in order to launch any of the hijacking attacks described earlier. This implies that off path attackers can no longer perform the hijacking attacks and that only on-path attackers can do so, so one may consider that a cookie based approach to secure MPTCP

connection results in similar security than current TCP. While it is close, it is not entirely true.
The main difference between the security of a MPTCP protocol secured through cookies and the current TCP protocol are the time shifted attacks. As we described earlier, a time shifted attack is one where the attacker is along the path during a period of time, and then moves away but the effects of the attack still remains, after the attacker is long gone. In the case of a MPTCP protocol secured through the usage of cookies, the attacker needs to be along the path until the cookie is exchanged. After the attacker has learnt the cookie, it can move away from the path and can still launch the hijacking attacks described in the previous section.
There are several types of approaches that provide some protection against hijacking attacks and that are vulnerable to some forms of time-shifted attacks. We will next present some general taxonomy of solutions and we describe the residual threats:

*Cookie-based solution: As we described earlier, one possible approach is to use a cookie, that is sent in clear text in every MPTCP control message that adds a new address to the existing connection. The residual threat in this type of solution is that any attacker that can sniff any of these control messages will learn the cookie and will be able to add new addresses at any given point in the lifetime of the connection. Moreover, the endpoints will not detect the attack since the original cookie is being used by the attacker. Summarizing, the vulnerability window of this type of attacks includes all the flow establishment exchanges and it is undetectable by the endpoints.

*Shared secret exchanged in plain text: An alternative option that is more secure than the cookie based approach is to exchange a key in clear text during the establishment of the first subflow and then validate the following subflows by using a keyed HMAC signature using the shared key. This solution would be vulnerable to attackers sniffing the message exchange for the establishment of the first subflow, but after that, the shared key is not transmitted any more, so the attacker cannot learn it through sniffing any other message. Unfortunately, in order to be compatible with NATs (see analysis below) even though this approach includes a keyed HMAC signature, this signature cannot cover the IP address that is being added. This means that this type of approaches are also vulnerable to integrity attacks of the exchanged messages. This means that even though the attacker cannot learn the shared key by sniffing the subsequent subflow establishment, the attacker can modify the subflow establishment message and change the address that is being added. So, the vulnerability window for confidentially to the shared key is limited to the establishment of the first subflow, but the vulnerability window for integrity attacks still includes all the subflow establishment exchanges. These attacks are still undetectable by the endpoints. The SCTP security falls in this category.

* Strong crypto anchor exchange. Another approach that could be used would be to exchange some strong crypto anchor while the establishment of the first subflow, such as a public key or a hash chain anchor. Subsequent subflows could be protected by using the crypto material associated to that anchor. An attacker in this case would need to change the crypto material exchanged in the connection establishment phase. As a result the vulnerability window for forging the crypto anchor is limited to the initial connection establishment exchange. Similarly to the previous case, due to NAT traversal considerations, the vulnerability window for integrity attacks include all the subflow establishment exchanges. Because the attacker needs to change the crypto anchor, this approach are detectable by the endpoints, if they communicate directly.

6.3. NAT considerations

TOC

In order to be widely adopted MPTCP must work through NATs. NATs are an interesting device from a security perspective. In terms of MPTCP they essentially behave as a Man-in-the-Middle attacker. MPTCP security goal is to prevent from any attacker to insert their addresses as valid addresses for a given MPTCP connection. But that is exactly what a NAT does, they modify the addresses. So, if MPTCP is to work through NATs, MPTCP must accept address rewritten by NATs as valid addresses for a given session. The most direct corollary is that the MPTCP messages that add addresses in the implicit mode (i.e. the SYN of new subflows) cannot be protected against integrity attacks, since they must allow for NATs to change their addresses. This rules out any solution that would rely on providing integrity protection to prevent an attacker from changing the address used in a subflow establishment exchange. This implies that alternative creative mechanisms are needed to protect from integrity attacks to the MPTCP signaling that adds new addresses to a connection. It is far from obvious how one such creative approach could look like at this point.
In the case of explicit mode, you could protect the address included in the MPTCP option. Now the question is what address to include in the MPTCP option that conveys address information. If the address included is the address configured in the host interface and that interface is behind a NAT, the address information is useless, as the address is not actually reachable from the other end so there is no point in conveying it and even less in securing it. It would be possible to envision the

usage of NAT traversal techniques such as STUN to learn the address and port that the NAT has assigned and convey that information in a secure. While this is possible, it relies on using NAT traversal techniques and also tools to convey the address and the port in a secure manner.

7. Recommendation

TOC

The presented analysis shows that there is a tradeoff between the complexity of the security solution and the residual threats. In order to define a proper security solution, we need to assess the tradeoff and make a recommendation. After evaluating the different aspects in the MPTCP WG, our conclusion are as follows:
MPTCP should implement some form of reachability check using a random nonce (e.g. TCP 3-way handshake) before adding a new address to an ongoing communication in order to prevent flooding attacks.
The default security mechanisms for MPTCP should be to exchange a key in clear text in the establishment of the first subflow and then secure following address additions by using a keyed HMAC using the exchanged key.
MPTCP security mechanism should support using a pre-shared key to be used in the keyed HMAC, providing a higher level of protection than the previous one.
A mechanism to prevent replay attacks using these messages should be provided e.g. a sequence number protected by the HMAC.
The MPTCP protocol should be extensible and it should able to accommodate multiple security solutions, in order to enable the usage of more secure mechanisms if needed.

8. Security Considerations

TOC

This note contains a security analysis for MPTCP, so no further security considerations need to be described in this section.

9. IANA Considerations

TOC

This document does not require any action from IANA.

TOC

10. Contributors

Alan Ford - Roke Manor Research Ltd.

11. Acknowledgments

TOC

Rolf Winter, Randall Stewart, Andrew McDonald, Michael Tuexen, Michael Scharf, Tim Shepard, Yoshifumi Nishida, Lars Eggert, Phil Eardley, Jari Arkko, David Harrington, Dan Romascanu, Alexey Melnikov reviewed an earlier version of this document and provided comments to improve it.
Mark Handley pointed out the problem with NATs and integrity protection of MPTCP signaling.
Marcelo Bagnulo is partly funded by Trilogy, a research project supported by the European Commission under its Seventh Framework Program.

12. References

TOC

12.1. Normative References

TOC

[RFC0793]	Postel, J., “Transmission Control Protocol,” STD 7, RFC 793, September 1981 (TXT).

12.2. Informative References

TOC

[RFC4225]	Nikander, P., Arkko, J., Aura, T., Montenegro, G., and E. Nordmark, “Mobile IP Version 6 Route Optimization Security Design Background,” RFC 4225, December 2005 (TXT).
[RFC4218]	Nordmark, E. and T. Li, “Threats Relating to IPv6 Multihoming Solutions,” RFC 4218, October 2005 (TXT).
[RFC3972]	Aura, T., “Cryptographically Generated Addresses (CGA),” RFC 3972, March 2005 (TXT).
[RFC5062]	Stewart, R., Tuexen, M., and G. Camarillo, “Security Attacks Found Against the Stream Control Transmission Protocol (SCTP) and Current Countermeasures,” RFC 5062, September 2007 (TXT).

[RFC5535]	Bagnulo, M., “Hash-Based Addresses (HBA),” RFC 5535, June 2009 (TXT).
[RFC3775]	Johnson, D., Perkins, C., and J. Arkko, “Mobility Support in IPv6,” RFC 3775, June 2004 (TXT).
[RFC5533]	Nordmark, E. and M. Bagnulo, “Shim6: Level 3 Multihoming Shim Protocol for IPv6,” RFC 5533, June 2009 (TXT).
[RFC4960]	Stewart, R., “Stream Control Transmission Protocol,” RFC 4960, September 2007 (TXT).
[I-D.ietf-mptcp-multiaddressed]	Ford, A., Raiciu, C., and M. Handley, “TCP Extensions for Multipath Operation with Multiple Addresses,” draft-ietf-mptcp-multiaddressed-02 (work in progress), October 2010 (TXT).

Author's Address

TOC

	Marcelo Bagnulo
	Universidad Carlos III de Madrid
	Av. Universidad 30
	Leganes, Madrid 28911
	SPAIN
Phone:	34 91 6248814
Email:	marcelo@it.uc3m.es
URI:	http://www.it.uc3m.es