INTERNET-DRAFT
Intended Status: Informational
Expires: May 15, 2015

E. Lopez
Fortinet

November 11, 2014

Multipath TCP Middlebox Behavior
draft-lopez-mptcp-middlebox-00

Abstract

As implementation of MPTCP continues to grow, there will be interaction concerns regarding MPTCP sessions relative to the functionality of middleboxes, particularly those focused on network-based security. The purpose of this draft is to review this interaction of MPTCp sessions and middleboxes, the likely response of middlebox providers in dealing with any functional degradation due to MPTCP, and the potential requirements to support proxy functionality for MPTCP sessions.

Table of Contents

1 Introduction 3
1.1 Terminology 3
2. Impact of MPTCP on Middleboxes 3
2.1 Single-Session Bias 4
2.2 Middlebox Function Degradation 4
3 Independent Responses By Middlebox Providers 5
3.1 MPTCP Fallback to TCP 5
3.2 Fast Closure of Existing MPTCP Sessions 5
3.3 Development of MPTCP-Aware Middlebox Functions 6
3.4 Independent TCP<>MPTCP Edge Proxies 6
3.5 MPTCP Third-Party Proxies 7
4 Security Considerations 7
5 IANA Considerations 8
6 References 8
6.1 Normative References 8
6.2 Informative References 8
Authors' Addresses 8

1 Introduction

Multipath TCP (MPTCP) as described in RFC 6824 [RFC6824], provides for end-to-end support for sessions utilizing multiple TCP subflows to allow MPTCP sessions to benefit from the available performance increase and persistence of using multiple forwarding paths. A difficulty comes from the deployment of middleboxes performing traffic inspection functions that are unaware of the multipath nature of the overall session, as well as suffering from a lack of visibility to all of the data within the complete session. In either case, the functionality offered by the middlebox is degraded in the presence of MPTCP sessions, where this was not the case for TCP sessions.

Much of this is based on the what could be termed the 'single-session bias' on the part of middlebox functionality. This means that the majority of functions supported by middleboxes, relative to TCP, assume that the information required by the function exists within a single TCP session. Clearly, the evolution of MPTCP invalidates this assumption or single-session bias. For example, a middlebox performing a network security function, such as intrusion prevention system (IPS), may not see all of the traffic required to match a signature within a single TCP subflow, even though the intrusion data is present within the overall MPTCP session. Even if this example IPS does happen to trigger on an intrusion, its actions would be likely be limited to the singular TCP subflow, rather than the overall MPTCP session.

The purpose of this draft is discuss the impact of MPTCP sessions on middlebox operation, with an attempt to understand how middleboxes will respond to the growing presence of MPTCP traffic. The evolution of support for MPTCP on middleboxes will result in requirements to support proxy functions to allow the data from all TCP subflows associated with an MPTCP session to be examined at an aggregation point on the network.

1.1 Terminology

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].

2. Impact of MPTCP on Middleboxes

Overall MPTCP sessions are an aggregation of one or more associated TCP subflows. Early implementations of MPTCP are generally

associated with establishing sessions between endpoint devices, while intermediate network devices handling the TCP subflows do not require awareness of the overall MPTCP session to perform packet forwarding decisions. The existence and utilization of multiple forwarding paths, and the use of different IP source and destination addresses on each subflow is inconsequential to the network, as it is the MPTCP endpoints that are responsible for the overall MPTCP session integrity.

However, intermediate network nodes, i.e. middleboxes, may perform functions on traffic other than forwarding. Such functions, if performed independently on a single TCP subflow, may provide erroneous or skewed results, as well as negatively impact the integrity of the overall MPTCP session. A strong example of an erroneous middlebox result would be the resulting false-negatives due to failures in signature-matching functions, since the matching data is distributed across multiple TCP subflows. A strong example of impact to the integrity of the MPTCP session would be the imposition of rate-limiting or packet transformation functions on the data contained within a single TCP subflow, and subsequent requirements placed on the MPTCP speakers to detect a degraded subflow.

2.1 Single-Session Bias

Most middleboxes suffer from what can be termed a 'single-session bias', in which each session is considered a unique data transfer between two endpoints, such that the functions being performed by the middlebox are bounded to the data within that single session. In the case of TCP, sessions are seen as unique, rather than an element of an overall MPTCP subflow. Even if a middlebox were to have visibility to all of the TCP subflows associated with an MPTCP session, each subflow would be segregated from the others by a differential set of source and/or destination IP addresses, as well as by different TCP sequence values. Without an understanding of MPTCP, MPTCP sessions are impacted by the interaction of middleboxes on individual TCP subflows.

MPTCP has been designed to be tolerant of NAT functions performed by middleboxes. However, other packet transformations, such as manipulation of DiffServ values or payload substitutions, can have very unintended effects of the overall MPTCP session. The issuance of a TCP reset (RST) by a middlebox only fast closes the individual TCP subflow, and not the overall MPTCP session. Ultimately, the solution to overcoming single-session bias effect will require evolution of middleboxes that are MPTCP-aware.

2.2 Middlebox Function Degradation

The functionality provided by middleboxes is too large in scope to cover within a single document. However, we can extropolate possible avenues for their functional degradation and failure , based on understanding the nature of MPTCP.

Since data within an MPTCP session can make use of multiple paths, a middlebox on any single path can suffer from a lack of complete visibility to the overall MPTCP session traffic. The likely outcome of this lack of visibility is a failure of middlebox function due to false-negative conditions.

A grave issue would be in the substitution of data within a single TCP subflow, and its impact on the overall MPTCP session. There are no functions defined in which the integrity of data contained within one TCP subflow is validated by another subflow. Data substitution within a single TCP subflow impacts the integrity of the overall MPTCP session, and is a significant security issue relative to MPTCP session operation.

3 Independent Responses By Middlebox Providers

The likely outcome is that providers of middleboxes will initially view MPTCP traffic as an attack relative to their operation. It is of course desirable for middleboxes to evolve to become MPTCP aware, and even support future MPTCP proxy functions. The following subsections describe methods in which middleboxes may respond to the presence of MPTCP traffic.

3.1 MPTCP Fallback to TCP

The methodology for MPTCP session fallback to standard TCP is clearly defined in RFC-6824 [RFC6824], section 3.6. Therefore the conditions under which such fallback can occur become available actions under which a middlebox can react to the presence of MPTCP. By forcing fallback to standard TCP, the middlebox can effectively mitigate functional degradation issues associated with MPTCP.

This is easily accomplished in one of two ways. First is to drop TCP traffic that contains TCP Options for MPTCP. Second is to filter out TCP Options for MPTCP, and forward the packet without the options.

3.2 Fast Closure of Existing MPTCP Sessions

It may also occur that a middlebox, rather than detecting the start of an MPTCP session, may instead detect the creation of a new TCP subflow via a Join Connection (MP_JOIN) MPTCP option. The issuance of a TCP Reset (RST) would only affect this TCP subflow, but not the overall MPTCP session.

RFC-6824 [RFC6824] does allow for fast closure of an overall MPTCP session, via the MP_FASTCLOSE option. However, this option must be authenticated with the key of the host to which it is sent. A middlebox, with only visibility of an MP_JOIN would not have knowledge of the key credentials of either Host A or Host B to be able to masquerade an MP_FASTCLOSE option.

This is of course a trade-off. While middlebox providers may desire the ability to issue third-party MP_FASTCLOSE options to both MPTCP hosts, providing the ability to do so would present a significant security challenge. Middleboxes have little recourse when only in the path of secondary TCP subflows of an MPTCP session.

3.3 Development of MPTCP-Aware Middlebox Functions

Another area of advancement is in the development of MPTCP middlebox functions. Much of this effort would result in blacklisting/whitelisting of MPTCP-capable sites. For example, web filtering solutions may include information on if sites are MPTCP-capable, and offer middlebox operators the option to allow MPTCP sessions toward specific sets of sites, rather than in a general allow/block state.

Note that this assumes that the middlebox is in the path of the initial TCP subflow establishing the MPTCP session.

3.4 Independent TCP<>MPTCP Edge Proxies

When deployed close to endpoints, the development of TCP<>MPTCP proxies would allow middleboxes visibility to all traffic associated with an MPTCP seesion. If effect, the middlebox becomes the endpoint for the MPTCP session. This would require the development of an explicit proxy function to convert standard TCP sessions from endpoints into an MPTCP session from a multipath-capable middlebox.

The Internet-Draft 'draft-wei-mptcp-proxy-mechanism-00' [WEI] provides an in-depth description of how such independent TCP<>MPTCP proxies could function. However, the use of such edge-proxies assume that:

- The MPTCP edge proxy is in the primary forwarding path between endpoints- The MPTCP edge proxy would force fallback to TCP for MPTCP capable endpoints behind the proxy

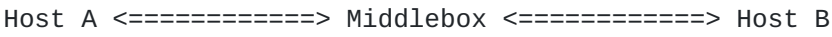
This would be necessary to prevent potential failures due to nested MPTCP sessions when both endpoints and middleboxes are MPTCP capable.

3.5 MPTCP Third-Party Proxies

While the development of MPTCP edge proxies is relatively straightforward, their deployment does not cover the cases under which a third-party can observe traffic from all TCP sub-flows for a given MPTCP session. It is not the intention of this draft to speculate on why it would be desirable to allow a third-party to perform such an aggregation of TCP subflows, and certainly from the standpoint of the MPTCP endpoints, such an aggregation is likely sub-optimal to the performance and resiliency of MPTCP sessions. However, the intent of this section regards what a third-party middlebox could do.

Two distinct models for a third-party proxy can be described: an in-path model, and an out-of-path model.

The in-path model is in essence an MPTCP<>MPTCP proxy, in which a middlebox inserts itself as a man-in-the-middle (MITM) between two MPTCP endpoints



Within the in-path model, it could be assumed that the Middlebox could adjust the initiation of an MPTCP session, for example by modification of a DNS Reply message, indicating that Host B is masqueraded by an IP address on the Middlebox. Host A then initiates an MPTCP proxy to the Middlebox, which in turn by its proxy function then establishes a separate MPTCP session to Host B.

The out-of-path model assumes that MPTCP capable endpoints, or downstream devices, will encapsulate their traffic, or a copy of their traffic, to a third-party middlebox. Traffic encapsulation can be used to differentially forward packets to a third-party. In this case, the third-party middlebox is assumed to be transparent to the MPTCP session establishment between endpoints.

Note that the MPTCP endpoints do not need to explicitly negotiate such a proxy, and may not even be aware of such a proxy taking place.

4 Security Considerations

This draft is fully concerned about the integrity of MPTCP sessions as they traverse middleboxes imposed in their path. While the deployment of middleboxes may in fact be benevolent in intent or practice, such devices may currently suffer from a single-session

bias relative to TCP subflows in an overall MPTCP session, which can cause erroneous or degraded functionality, with potential impact on the overall MPTCP session. As middleboxes become MPTCP aware, the rise of independent and third-party proxy functions may exploit MITM weaknesses available within and external to the MPTCP protocol.

5 IANA Considerations

There are no new requirements for IANA consideration in this draft. However, 'draft-wei-mptcp-proxy-mechanism-00' [WEI] suggests that a new flag 'P' in MPTCP MP_CAPABLE option needs to be defined, refer to RFC 6824, Section 3.1. This flag could be used by a proxy to inform MPTCP capable host the existence of proxy, although as this draft suggests such proxies can we created without informing the MPTCP capable hosts of their presence.

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, January 2013.

6.2 Informative References

[WEI] Wei.X, Xiong, C. "MPTCP proxy mechanisms", draft-wei-mptcp-proxy-mechanism-00, June 2014

Authors' Addresses

Edward Lopez
Fortinet
899 Kifer Road
Sunnyvale, CA 94086

EMail: elopez@fortinet.com