MULTIPATH TCP DEPLOYMENTS

By Olivier Bonaventure and SungHoon Seo

MULTIPATH TRANSMISSION CONTROL PROTOCOL (MPTCP), SPECIFIED IN RFC 6824[1], is the most recent extension to the venerable TCP. TCP was designed when hosts had a single network interface and a single IP address. Each TCP connection is identified by a four-tuple (source and destination addresses and source and destination ports), and every packet belonging to this connection carries this four-tuple. Once a TCP connection has been established, it is impossible to change any of the elements of the four-tuple without breaking the connection—a severe limitation in today's networks for the following reasons:

1. Many hosts are dual-stack. Even if they have a single interface, they have two or more addresses and there are different network paths between any pair of communicating hosts.
2. Many hosts have several interfaces, such as smartphones and tablets.
3. There are a growing number of mobile hosts on today's Internet with addresses that can change as they move from one wireless network to another.

Multipath TCP handles these issues by extending TCP to enable endhosts to exchange data belonging to one connection over different paths. To achieve this, Multipath TCP combines several TCP connections (called subflows in RFC 6824) into a single Multipath TCP connection. The first subflow starts with a three-way handshake, much like a regular TCP connection. The main difference is that the SYN packet contains an MP_CAPABLE option that negotiates the use of Multipath TCP and random keys.

Once the first subflow has been established, either of the communicating hosts can create an additional subflow from any of its own addresses toward any of the addresses of the remote host by sending a new SYN with the MP_JOIN option. Such subflows can be created and terminated at any time, which is very important for mobile hosts. Data can be sent over any of the subflows that currently compose the Multipath TCP connection. If a subflow fails, all the data that was transmitted over it that has not yet been acknowledged will be retransmitted over other subflows. (For more information about Multipath TCP, see RFC 6824 or NSDI '12[2]).

Today there exist multiple independent interoperable implementations of Multipath TCP. The most widely used are iOS/macOS and Linux[3]. Multipath TCP is supported by load balancers, and there are implementation efforts on FreeBSD and Solaris. This article describes several commercial services that leverage the unique capabilities of Multipath TCP.

Smartphones

The largest deployment of Multipath TCP is on smartphones.

End-to-end Multipath TCP

Smartphones often have connectivity to both a WiFi access point and a cellular network. If a user has Internet connectivity via WiFi, walking away from the WiFi access point will result in the smartphone losing connectivity, implying that the TCP connection that has been established over WiFi will also fail. One of the benefits of Multipath TCP is its ability to seamlessly hand over from one interface to another—making it the perfect candidate to solve the these kind of losses of connectivity (Figure 1).

Siri is the digital assistant in Apple's iOS and macOS operating systems. Because speech recognition requires tremendous processing power, Siri streams spoken commands to Apple's datacenter for speech recognition; the result is sent back to the smartphone. Although the duration of a user's interaction with Siri is relatively short, Siri's usage pattern made this data transfer a perfect client for MPTCP.

Many people use Siri while walking or driving. As they move farther away from a WiFi access point, the TCP connection used by Siri to stream its voice eventually fails, resulting in error messages.

To address this issue, Apple has been using MPTCP—and benefiting from its

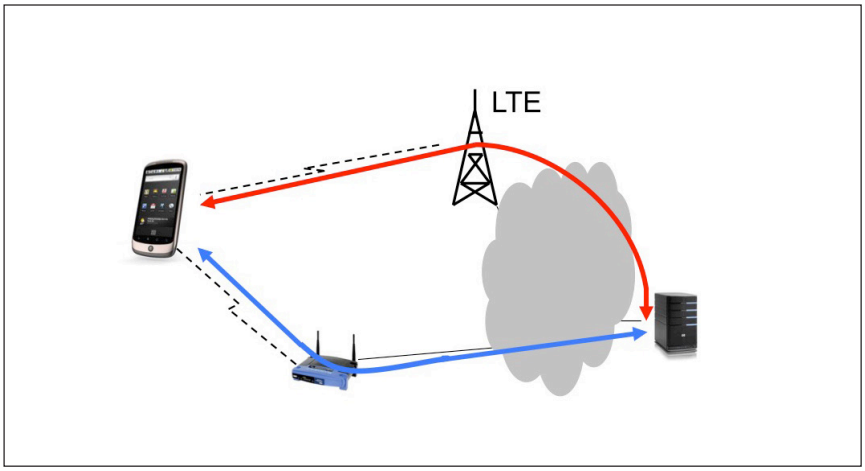

Figure 1. Illustration of Taking Advantage of Multiple Interfaces

handover capabilities—since its iOS 7 release. When a user issues a Siri voice command, iOS establishes an MPTCP connection over WiFi and cellular. If the phone loses connectivity to the WiFi access point, traffic is handed over to the cellular interface. A WiFi connection that is still in sight of an access point can have a channel become so lossy that barely any segments can be transmitted. In this case, another retransmission timeout happens and the iOS retransmits the traffic over the cellular link.

Deploying MPTCP on the Internet has been relatively painless. MPTCP's ability to handle middlebox interference and fall back to regular TCP has proven efficient and without major issues.

To further reduce latency, iOS measures the round-trip times (RTTs) on the two interfaces. Bufferbloat is infamously known to cause huge RTTs. The WiFi link may have an RTT much bigger than that of the cellular link. When iOS detects that the RTT over WiFi is much bigger than the one over cellular, it sends the voice stream over the cellular interface.

Finally, input from WiFi Assist[4] and a trigger to hand over traffic to the cellular interface is used. For Siri users, this deployment of MPTCP has resulted in a significant reduction of network errors. After establishing two subflows (one over WiFi and one over cellular), the network error rate decreased by 80%.

Thanks to RTT measurements that also trigger handovers, Siri also responds faster to user commands. Siri can provide user feedback 20% faster in the 95th percentile and 30% faster in the 99th percentile.

Deploying MPTCP on the Internet has been relatively painless. MPTCP's ability to handle middlebox interference and fall back to regular TCP has proven efficient and without major issues. Roughly 5% of the connections, however, still fall back to regular TCP, due to both the deployment of transparent TCP proxies in cellular networks and firewalls removing MPTCP options.

One challenge of MPTCP is its debuggability. Subflow handling introduces a major code complexity: WiFi interfaces appear and disappear. Some of these networks may have middleboxes that interfere with MPTCP, making subflow establishment impossible. Corner-case scenarios, which are hard to reproduce and only happen when a product is deployed at huge scale, require extensive logging mechanisms to trace the behavior of an MPTCP connection.

Due to the uncertainties introduced by middleboxes on a network, it is difficult to identify root cause of an issue. As a result, one can't always differentiate between a software bug and a middlebox.

Multipath TCP through SOCKS proxies

Besides the servers deployed specifically for the previous use case, there are very few servers that already support Multipath TCP. Despite this, several network operators seek to enable smartphone users to achieve increased throughput by combining existing cellular and WiFi networks. Network operators in several countries have relied on SOCKS (RFC 1928[5]) to simultaneously use WiFi and cellular networks. From an operator's viewpoint, the main benefit of coupling SOCKS with MPTCP is that it is easily deployable, since no or few dependencies exist with the existing cellular core and WiFi infrastructure.[6]

From an operator's viewpoint, the main benefit of coupling SOCKS with MPTCP is that it is easily deployable, since no or few dependencies exist with the existing cellular core and WiFi infrastructure.

Several models of commercial Android smartphones include the Multipath TCP implementation in the Linux kernel and a SOCKS client. The SOCKS client running on the smartphone intercepts any TCP connection attempts to distant servers. It then creates a connection to a SOCKS server managed by the network operator.

When the user is authenticated, the SOCKS client sends a command to the SOCKS server, which creates a TCP connection toward the remote server. At this point, there is a Multipath TCP connection between the smartphone and the SOCKS server, and a TCP connection between the SOCKS server and the remote server. The SOCKS server relays all data sent on the Multipath TCP connection over the TCP connection, and vice versa. Smartphones create additional subflows toward the SOCKS server over the other available interfaces. The result is an improved user

Continued on next page

Multipath TCP Deployments, continued

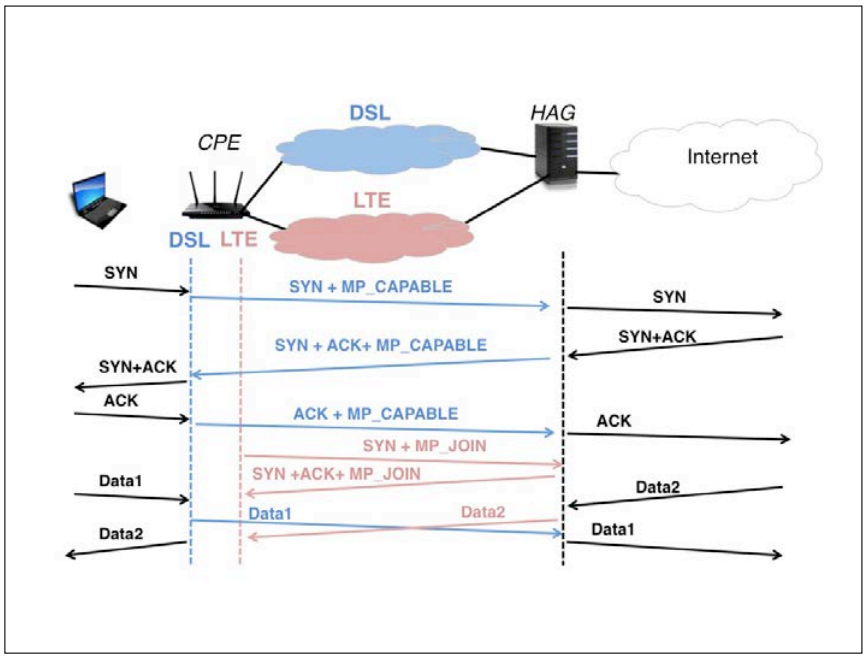

Figure 2. The Multipath TCP Handshake

experience, thanks to aggregated bandwidth and seamless handover.

Hybrid Access Networks

Another important use case for Multipath TCP lies in access networks. In many regions of the world, the available access networks provide limited bandwidth. A typical example is rural areas, where it is costly for network operators to deploy high-bandwidth access networks. Even if access network bandwidth is limited, it often is possible to subscribe to different network services that, when combined, provide higher bandwidth and higher resiliency.

Several companies have deployed solutions that leverage the unique bonding capabilities of Multipath TCP. The first relies on SOCKS proxies and enables endusers to efficiently combine network services from different providers. The second is targeted at network operators seeking to combine fixed (e.g., xDSL) and wireless (e.g., LTE) networks in order to provide higher bandwidth to customers.[7]

Even if access network bandwidth is limited, it often is possible to subscribe to different network services that, when combined, provide higher bandwidth and higher resiliency.

Combining access networks with SOCKS

SOCKS is also used with Multipath TCP to combine different access networks. In this deployment, end hosts are regular hosts that do not support Multipath TCP. To benefit from the bonding capabilities of Multipath TCP, a middlebox is installed in the end user's LAN. This middlebox acts as a SOCKS client and interacts with a server in the cloud. Both the middlebox and the cloud server both use Multipath TCP and, therefore, are able to exploit any available access network, provided an IP address has been assigned to the middlebox on each of the access networks.

The middlebox typically acts as a default gateway in the end user's LAN. It intercepts all TCP packets sent by the hosts on the LAN to external destinations, and then it proxies them over Multipath TCP connections toward a SOCKS server running in the cloud. This server terminates the Multipath TCP connections and initiates regular TCP connections to the final destinations.

This solution is already commercially deployed in two countries. Users report successfully combining different types of access links, including xDSL (from ADSL to VDSL), DOCSIS, 3G, 4G, and satellite links.

Multipath TCP in hybrid access networks

Some network operators have deployed both fixed (e.g., xDSL) and wireless (e.g., LTE) networks and wish to combine the networks in order to offer higher bandwidth services. Multipath TCP may also be used to provide these services (Figure 2).

In this deployment, neither the client nor the server support Multipath TCP. Multipath TCP is used on the CPE and in the hybrid aggregation gateway (HAG) that resides in a datacenter of the network operator that manages both access networks[8].

When a client initiates a TCP connection toward a remote server, it sends a SYN packet. This packet is intercepted by the CPE that virtually terminates the TCP connection and then adds the MP_CAPABLE TCP option before forwarding the packet over the xDSL network. The HAG, which resides on the path followed by all packets sent by the client over the xDSL network,

intercepts the SYN packet. It virtually terminates the Multipath TCP connection and then forwards the SYN to the server after having removed the MP_CAPABLE option. The server then confirms the establishment of the connection by sending a SYN+ACK. This packet is intercepted by the HAG that updates its state for this connection and adds an MP_CAPABLE option before forwarding it toward the CPE. The CPE performs similar operations. It updates its state and forwards the SYN+ACK to the client without the MP_CAPABLE option to confirm the establishment of the connection.

At this point, there are three TCP connections. The first is a regular TCP connection. It starts at the client and is virtually terminated on the CPE. The second is a Multipath TCP connection that is virtually terminated on the CPE and the HAG. And the third is a regular TCP connection between the HAG and the remote server. From an operational viewpoint, it is important to note that with IPv6, neither the CPE nor the HAG need to translate the source and destination addresses of the forwarded TCP packets. The client IP address remains visible to the destination server. This is an important advantage compared to SOCKS-based solutions.

Despite its young age, Multipath TCP is deployed on a large scale for several commercial services.

Furthermore, in this deployment the connection between a client and a server can be created within a single round-trip time.

Conclusion

Despite its young age, Multipath TCP is deployed on a large scale for several commercial services. On smartphones, it combines cellular and WiFi networks both for higher bandwidth and for faster handovers in delay-sensitive applications. In access networks, it supports hybrid access networks that improve customer experience by efficiently combining existing fixed and wireless networks.

Acknowledgements

The authors thank Christoph Paasch and Simon Lelievre.

References

1. Ford, A., Raiciu, C., Handley, M., and Bonaventure, O., “TCP Extensions for Multipath Operation with Multiple Addresses”, RFC 6824, DOI 10.17487/RFC 6824, January 2013.
2. Raiciu, C., Paasch, C., Barré, S., Ford, A., Honda, M., Duchêne, F., Bonaventure, O., and Handley, M., “How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP”, USENIX Symposium of Networked Systems Design and Implementation (NSDI ’12), April 2012.
3. Paasch, C., Barré, S., et al. “Multipath TCP implementation in the Linux kernel”, https://www.multipath-tcp.org.
4. Apple, WiFi Assist, https://support.apple.com/en-us/HT205296.
5. Leech, M., Ganis, M., Lee, Y., Kuris, R., Koblas, D., and Jones, L., “SOCKS Protocol Version 5”, RFC 1928, March 1996.
6. S. Seo, “KT’s GiGA LTE”, IETF 93, https://www.ietf.org/proceedings/93/slides/slides-93-mptcp-3.pdf.
7. Broadband Forum, TR 348–Hybrid Access Broadband Network Architecture, July 2016, https://www.broadband-forum.org/technical/download/TR-348.pdf.
8. B. Peirens, G. Detal, S. Barré, O. Bonaventure, “Link bonding with transparent Multipath TCP”, Internet-Draft, draft-peirens-mptcp-transparent-00, 2016.