Simulation-based study of MPTCP (Multipath TCP)

Bachir Chihani, Denis Collange

A Survey on Multipath Transport Protocols

Bachir Chihani - Denis Collange
Orange Labs Sophia Antipolis, France
Email: bachir.chihani,denis.collange@orange-ftgroup.com

***Abstract*—We present a survey on multipath transport protocols. These protocols are aiming to provide a way for the use of simultaneous paths at the transport layer and load balancing traffic on these paths. We describe some of the main proposal and then we focus on MPTCP (Multipath TCP) which is a promising extension of TCP currently considered by the recent eponymous IETF working group.**

***Index Terms*—Multipath transport protocols, Congestion control, Load balancing, Scheduling.**

I. Introduction

Nowadays mobile equipment have often more than one single network interface. For instance, laptops have usually at least both a wired (Ethernet) and a wireless (Wifi) network adapters. Similarly smartphones and tablet PCs can reach the Internet either through Wifi or through a cellular network (UMTS or 3G+).

Another fact is that operators usually duplicate links and network equipments to protect their networks against failures, especially in their access and backhaul networks. Moreover the backbone networks are generally meshed. In this context many paths can exist between any two endpoints. The idea to use concurrently many paths has then emerged, in order to improve the robustness and performance of end-to-end connections. Such multipath connections can indeed balance the load between the different paths, switch dynamically the traffic to the best path, avoiding congested or broken links.

A lot of studies have considered the implementation of multipath capabilities at different layers: at the application layer [5], at the transport layer [6], [7], [1], [12], [13], [14], [7], etc.

We think that it is at the transport layer where end-systems can make maximum benefit of the multipath [8]. At this layer, end-systems can gather information about each used path: capacity, latency, congestion state. These information can then be used to react to congestion in the network by moving the traffic away from congested paths.

An IETF's working group has recently been created to standardize a multipath protocol at the transport layer. They proposed Multipath TCP [9] (MPTCP), an extension of TCP to handle multiple paths between two endpoints.

The reminder of this paper is as follow: we first present in section II some protocols handling multipath at the transport layer. Then we describe MPTCP in section III, as it is specified in the current versions of the IETF drafts. In section IV, we discuss the implemented mechanisms in the different cited protocols. Finally, we conclude the paper in section V.

II. Multipath transport protocols

A. ATLB (Arrival-Time matching Load-Balancing)

ATLB [7] is a transport protocol supporting the multipath. It allows the distribution of data among different available paths with the objectives to minimize the dis-sequencing of packets at the receiver side, the detection of problems on paths, and the recovery of lost packets. ATLB provides a way to mesure the arrival time of packet to the receiver. It defines this time as the queuing delay plus the network delay (smoothed Round-Trip-Time). Using these delays, ATLB attributes a score for each path and use the path with the least score to send data.

$$score_i = \frac{Q_i}{G_i} + \frac{sRTT_i}{2}$$

Q is the length of data in the sending buffer; $sRTT$ is the smoothed Round-Trip-Time; G is a smoothed throughput computed as $G_j = \alpha * G_j - 1 + (1-\alpha) * TPUT_j$. $\alpha (0 < \alpha < 1)$ is a constant, and $TPUT_j$ is the throughput of a TCP connection measured each β milliseconds.

For path failure detection, ATLB maintains a timer which expires after a time T. ALTB assumes that a lost segment after an RTO expiration means a path failure or a path highly congested. Thus $T = RTO + \theta * RTT (\theta > 2)$.

In case of path failure, ATLB sends a sensing packet each P ms. It also compute the lost rate each S ms. If the lost rate is less than $R\%$, then it assume that the packet is recovered and can be used.

B. Fair-TCP

Fair-TCP [2] is a protocol supporting the multipath, implemented at both the sender and the receiver side. It was conceived for SAN (Storage Area Network) where the TCP sessions are maintained for a long period of time and the exchanges are based on SCSI input/output commands, and which are part of a communication standard called iSCSI (Internet SCSI). For enhancing the performance of the communications, Fair-TCP shares congestion information between the different connections by the use of a data structure called the ECB (Ensemble Control Block) which is an extension of the TCP Control Block.

C. PATTHEL

PATTHEL [4] is a solution implemented at the session layer to paralelly transfert data. It provides an API (Application Programming Interface) for the application developper to use this protocol. The protocol uses a dedicated channel (end-to-end path) created in first for controlling the connection, the

rest channels are used to transfert data. A received data block from the application layer is divided into chunks of variable size depend of the channel characteristics. To each chunk a header is added which contain:

- Chunk size (32 bits) to indicate the size of the payload;
- Stream offset (64 bits) to indicate the position of the data on the flow;
- Block size (32 bits) allows the receiver to check if the block can be hold in the application buffer;
- Block index (32 bits) used to check if the chunk belong to the block currently in reception.

To force sending packets over a certain channel, PATTHEL add an entrance to the routing table.

D. R-MTP (Reliable Multiplexing Transport Protocol)

R-MTP [3] is to used by mobile nodes having many wireless interfaces of potentially heterogenous technologies. It is a transport protocol able to agregate the available bandwidth of different network paths by distributing data over these paths. The protocol maintain a set of information about each used path in order to react to any change happening over a path. Like the bandwidth which is estimated by the Packet Pair method. This one helps to estimate the least time interval between packets so that to avoid queuing delays. The protocol makes use of a special header format to exchange information between endpoints and SACK (Selective Acknowledgements) for reliability. Example of the exchanged information which can be included in the header:

- *Initial rate* is the reverse of the minimum period (between sending two packets) that the path can support without creating congestion;
- *Interarrival time* is the difference between the arrival time of the precedent packet and the current one, on the same path;
- *Jitter* is the difference between the measured interarrival time and the rate on the same path;
- *Commulative long run jitter* is the commulated jitter which in case it grows can be interpreted as a congestion.

E. cTCP (Concurrent TCP)

cTCP [1] is a TCP-based protocol which allows the use of multiple paths between two hosts having many network interfaces. Figure **??** illustrates the architecture of this protocol which is composed of a packet scheduler used for laod-balacing the traffic on the different paths; an acknowledgment processor used to fix the gap report problem in the TCP congestion control. This component include a Duplicated ACK classifier which can provide information about the quality of a path to the packet scheduler; An interne databases implemented as a chained linear list. To remain compatible with TCP standards versions, cTCP uses a single congestion window to control the global throughput and a single emission buffer shared between the different paths. At the connection establishment, the two enpoints exchange their available addresses using specific option, if one of the endpoint isn't a cTCP one it will ignore the options putting the connection to be a standard TCP one. The used packet scheduling algorithm is a Credit-Weighted Round-Robin. This one allows a fair data distribution among the different paths. Whanever an acknowledgement is received, the estimated bandwidth of the corresponding path is updated and a new sending credit is added to the sender. The new credit is then divided between the different paths.

For the congestion control, cTCP uses the database to store all unacked packet with the identifier of the path used to send the packet. When a new ackowledgement is received, the sender drop all packet having a sequence number less the one included in the acknowledgment. When a duplicated acknowledgment is received, the sender checks if the path used to send the packet suspected to be lost and the path from which the dupack has been recevied are the same. If it is, then a Fast Retransmit occurs, otherwise the dupack is ignored.

III. MULTIPATH TCP

An IETF's working group has been created to standardize a multipath protocol for the transport layer. They proposed MPTCP [9] (Multipath TCP), an extension of TCP to handle multiple paths between two endpoints. MPTCP is designed with three major goals:

1) **Improve throughput**: the performance of a multi-path flow should be at least as good as this of a single-path flow on the best route.
2) **Do no harm**: a multi-path flow should not take up any more capacity on any one of its paths than a single-path flow using that route.
3) **Balance congestion**: a multi-path flow should move as much traffic as possible away from the most congested paths.

A. Main mechanisms

With MPTCP, the transport layer is splitted in two sublayers. The upper one gathers the functionalities for connection management (establishing connection, reordering packets, etc.). The lower one gathers a set of subflows that can be seen as one TCP flow. MPTCP distinguishes two spaces for sequence numbers. Each subfow has its own sequence space which is similar to the Standard TCP sequence number, identifying bytes within a subflow. At the connection level, another sequence space is used for reordering purposes.

The MPTCP protocol use new TCP options to exchange signalling information between peers, for instance:

MPC (Multipath Capable) is used during the three-way handshake to establish a multipath TCP connection.

DATA FIN is used to inform the remote peer of the end of data and to close the multipath TCP connection.

ADD and **REMOVE** Address (Ipv4) are used to inform the remote peer of the availability of a new address or to ask it to ignore an existing one.

JOIN is used to initiate a new sub-flow (packet flow on a route) between a not used peer of addresses.

DSN (Data Sequence Number) is used as a map between subflow level and data sequence space number.

1) Connection establishment: Figure 1 illustrates the process of establishment of a MPTCP connection. After that the source application sends a Connect() call, the transport layer establishes a connection with the destination peer which was waiting for receiving connection requests. The establishment is TCP-like (three way handshake) with the use of MPC option to inform the other peer that the initiator is able to exchange data using multipath TCP. To initiate subflows, peers must first exchange their additional addresses. The MPCTP draft do not specify how the exchange may happen. We choosed to send additional TCP segments. These segments handle the ADDR (Add address) option and are sent just after successfully establishing the connection.

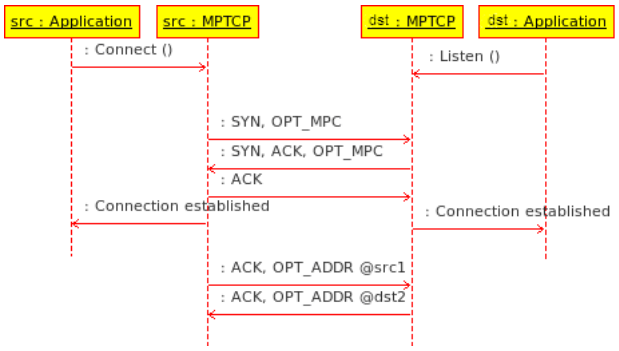

Fig. 1. MPTCP connection establishment

2) Subflow initiation: Figure 2 shows the initiation of a new subflow and the presence of a JOIN in a SYN segment. To maximize the chance that the subflow under initiation takes a path which is disjoined with previously established paths, an address is used only by a one subflow.

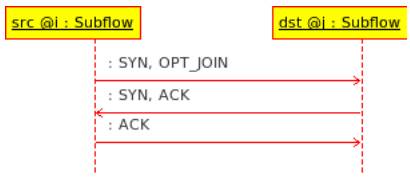

Fig. 2. MPTCP sub-flow initiation

B. Traffic control

MPTCP redefines some TCP mechanisms so that they fit the multi-path context. Congestion control allows the sender to regulate its throughput according to the available network resources.

In MPTCP, the congestion control is performed at the subflow level. Each subflow has its own congestion window. But the congestion windows of different subflows may be coupled to improve the performance.

In MPTCP, the receiver has only one global receiving window shared between the set of the established subflows. The objective is to do not limit the speed of some subflows.

Four different algorithms have been proposed by Raiciu et al. in [10], coupling in various ways the congestion windows of active subflows: Uncoupled, Fully Coupled, Linked Increase, and RTT Compensator. They consider a simple extension of TCP's congestion control in case where the round-trip time is the same for all the available paths $r = 1, ..., N$.

With the algorithm Uncoupled, the congestion window of each subflow behaves like for a single Standard TCP connection. Let w_r be the congestion window on path r, and $w = \sum_r w_r$

Algorithm **Fully Coupled**

- $w_r = w_r + \frac{1}{w}$ per ACK on path r
- $w_r = max(w_r - \frac{w}{2}, 1)$ per loss event on path r

Most of the time either one path or another is used with this algorithm, and rarely both. This phenomenon is called Flappiness. To reduce flappiness, authors proposed the following algorithm:

Algorithm **Linked Increases** [11]

- $w_r = w_r + \frac{a}{w}$ per ACK on path r
- $w_r = \frac{w_r}{2}$ per loss event on path r

In more general case where RTT (round-trip time) are not equal for the all paths, the authors adjust the precedent algorithm:

Algorithm **RTT Compensator**

- $w_r = w_r + min(\frac{a}{w}, \frac{1}{w})$ per ACK on path r
- $w_r = \frac{w_r}{2}$ per loss event on path r

IV. Discussion

Follow we describe the different mechanisms that a reliable and connection oriented transport protocol msut have them, also the modification which must occurs to these mechanisms for adapting them to the multipah.

A. Congestion control

For a multipath TCP connection, the congestion control must happen in each used paths so that we can match in real-time manner the quantity of data sent on a path and the capacity of this one. Also, a congestion control for a multipath TCP solution must satisfy the previously stated goals: *improve throughput, do no harm, balance congestion*. Like in TCP and for each path, the sender has to maintain a set of parameters for the congestion control and the updates after receiving acknowledgement and duplicated acknowledgement: RTT, RTO, ssthresh, cwnd, etc. The congestion control must happen on each path in takking into account its characteristics. It is foreseeable that a coupling strategy can be used to combin the differents paths parameters in order to aggregate the ressource of the available paths, or move away data from congested paths.

B. Flow control

Most of the proposed solutions for flow control in the multipath transport protocols are a simple adaptation of the TCP control flow. The receiver maintains a single window shared between all subflows in order to not limit the speed of some of them. The sender and the receiver agree on the awnd (advertised window) which represents the quantity of data which can be sent without exchanging acknowledgement.

The sender can then divides the window size between all the available paths according to the used data distribution policy.

C. Acknowledgement management

When receiving an acknowledgement on a path, the corresponding parameters are updated (round trip time, bandwidth, loss rate, congestion window). If some data need to be retransmitted, it is possible to do it on the same path used firstly to transmitted these data (even if that path is potentially congested), or on another one which is the preferable solution because the retransmission will be faster.

D. Loss packets recovery

In standard TCP, loss recovery happens after an RTO (Retransmission TimeOut) expiration, or the reception of three duplicated acknowledgment. The recovery consists of sending all not acked data. In a multipath context, the same mechanisms can be reused but after adaptation. For the duplicated acknowledgement we can vary the threshold, differentiate between spurious acknowledgements caused by the arrival of packets in out of sequence manner at the receiver, and the real acknowledgement stating a potential lost. For the RTO, for each used path needs a specific RTO so that when the RTO expires only the concerned path will be affected.

E. Failure management

To make sure that a path which wasn't used for a certain time is back operationnal, the protocol may use sensing message (like Heartbeat in SCTP) in defined interval. If all paths are used during a communication, the protocol may use a counter returned to zero each time some data are received and in case of expiration the path can be considered as fail.

F. Data distribution over paths

Most of the proposals use a Round Robin based sequencing policy, and distribute data fairly distributed. But paths have characteristics (latency, capacity, jitter, loss rate, etc.) which are potentially differents, a such policy is not intersting for multipath. Other protocols use an ameliorated Round Robin policy and distribute on each path an amount of data which is proportional to the throughput of the used path, ou only send data on the best path (like ATLB, M/TCP). Another proposal consist of sending data out of sequence and with proportional amount to the path characteristics so that data arrives in sequence to the receiver. A distribution method can take into account any combinaison of the parameters capacity, latency, jitter, loss rate, etc. But it has to ensure that data arrive in sequence without an overload a path while others are not used.

G. Managing out of sequence data arrival

Packets may take different paths, thus they may arrive at the receiver out of sequence. The receiver has to hold a free space to save these packets from the other which are makin a normal sequencing.

H. Path managment

Path management (i.e. adding or droping paths) may be done by using the option field of the TCP header. For choosing paths, these have to be disjoined which means they do not share the same physical link. Otherwise, in case of congestion or failure of one of them, all others will also be congested or fail. For the connection management, most of the presented protocols, even those no compatible to TCP, use the TCP's three way handshake.

V. Conclusion

References

[1] Yo Dong, N. Pissinou, and J. Wang. "Concurrency Handling in TCP." 5ème conférence annuelle en Communication Networks and Services Research (CNSR'07). 2007 IEEE.

[2] B. Kancherla, G. Narayan, K. Gopinath. "Performance Evaluation of Multiple TCP connections in iSCSI", 24th IEEE Conference on Mass Storage Systems and Technologies (MSST '07), 2007.

[3] L. Magalhaes and R. Kravets. "Transport level mechanisms for bandwidth aggregation on mobile hosts". ICNP (2001), 0165.

[4] A. Baldini, L. De Carli et F. Risso. "Increasing Performance of TCP Data Transfers Through Multiple Parallel Connections", IEEE Symposium on Computers and Communications (ISCC 09), Sousse, Tunisia, July 2009.

[5] T. Hacker and B. Athey, "The End-to-End Performance Effects of Parallel TCP Sockets on a Lossy Wide-Area Network ," in IEEE IPDPS, Florida, Apr. 2002.

[6] L. Ong, J. Yoakum, "An Introduction to the Stream Control Transmission Protocol (SCTP)," RFC 3286, May 2002.

[7] Y. Hasegawa, I. Yamaguchi, T. Hama, H. Shimonishi and T. Murase. "Improved Data Distribution for Multipath TCP communication," IEEE Globecom 2005.

[8] D. Wischik, M. Handley and M. Bagnulo Braun, "The Resource Pooling Principle", ACM SIGCOMM Computer Communication Review, Vol 38.5, pp 47-52, October 2008.

[9] A. Ford, C. Raiciu and M. Handley, "TCP Extensions for Multipath Operation with Multiple Addresses," draft-ietf-mptcp-multiaddressed-01, July 12, 2010.

[10] C. Raiciu, D. Wischik and M. Handley, "Practical Congestion Control for Multipath Transport Protocols," UCL Technical Report, 2009.

[11] C. Raiciu, M. Handley and D. Wischik, "Coupled Multipath-Aware Congestion Control", draft-raiciu-mptcp-congestion-01, March 2010.

[12] D. Sarkar."A Concurrent Multipath TCP and Its Markov Model," IEEE ICC proceedings, 2006.

[13] M. Zhang, J. Lai, A. Krishnamurthy, L. Peterson and R. Wang. "A transport layer approach for improving end-to-end performance and robustness using redundant paths," In ATEC'04: Proceedings of the annual conference on USENIX Annual Technical Conference (Berkeley, CA, USA, 2004), USENIX Association, pp. 8-8.

[14] K. Rojviboonchai and H. Aida. "An evaluation of multi-path Transmission Control Protocol (M/TCP) with robust acknowledgement schemes," Proc. Internet Conference 2002 (IC2002), Univ. Tokyo, Japan, October 2002.