Towards a Multipath Transport Protocol for Serval

Torin Rudeen
Department of Computer Science, Princeton University
Under the advisement of Michael J. Freedman

Abstract

Multipath transport protocols provide practical and theoretical advantages over traditional single path protocols, enabling better congestion balancing and higher throughput without violating TCP fairness conditions. Multipath TCP (MPTCP) is the main ongoing effort to create a multipath transport protocol, but its efficiency and flexibility is hampered by the need to be entirely backwards compatible with a network designed for single path connections. What would a multipath transport protocol look like if it could be created without these restrictions?

Serval is an experimental network architecture designed to support service-centric networking. It supports seamless mobility across networks and interfaces, multipath communication, and a service-centered network abstraction by creating a Service Access Layer which sits on top of an unmodified Network Layer and below the Transport Layer. In this paper I draw on the ongoing research on MPTCP to design and implement a multipath transport protocol that takes advantage of the abstractions provided by Serval to offer improved efficiency, flexibility, and simplicity, while still playing fair with legacy TCP connections.

1 The Need for Multipath

In his widely read article *The Rise of "Worse is Better"*[5], Richard Gabriel offered a novel theory of why technically imperfect software systems often gain traction and widespread acceptance, even where more sophisticated systems fail. Consider two systems: the first is simple to implement, but sacrifices completeness, consistency, and sometimes even correctness in service of this simplicity. The second is created to be completely correct, consistent, and complete, but sacrifices simplicity to this end. The first system is considerably "worse" than the second: it is unreliable, limited in what it can do, and imposes extra burdens on the user. However, it will be created faster, ported more easily, and by the time the second system has been perfected the first will already have been adopted by everybody. C is an example of the first type of system; Lisp is an example of the second. C was so successful not because it was technically superior to Lisp, but because it was much easier to write compilers for.

The Internet is one of the best examples of a "Worse is Better" system. Crucial to the widespread adoption of the Internet is the simplicity of the core protocols. Each piece of functionality is shunted as far up the network stack as possible: for example, instead of relying on the Link Layer or the Network Layer to guarantee reliable delivery, it's left to the Transport Layer. While this may be inefficient, because packets must be sent across the entire network when a loss occurs on any link, it makes switches and routers far simpler and cheaper to implement. Similarly, instead of explicitly allocating bandwidth to each connection, as in telephone networks, the Internet uses statistical multiplexing, relying on Transport level congestion avoidance algorithms to ensure an approximately fair allocation of resources.

However, while "Worse is Better" may be sufficient most of the time,

2 Overview of Serval

The network stack, in particular the TCP and IP protocols, was designed around a host-centric abstraction, optimized for an Internet of immobile computers which use only a single network interface. This is a far cry from today's Internet, where computers are mobile, possess multiple heterogenous network interfaces, and wish to connect to web services which are split across thousands of computers around the world.

For example, IP routes packets based on their IP address, implicitly relying on the assumption that each party in communication is uniquely identified by a single IP address that does not change over time. Unfortunately, each component of this assumption fails in today's Internet: IP addresses do not uniquely a party, due to the paucity of (IPv4) addresses and the resulting prevalance of NATs; a single party often has more than one IP address, whether because it is a single device with multiple interfaces (i.e. a smartphone with 4G and WiFi), or because it is a large web service run on thousands of different machines (i.e. Google); and the IP address of one party can change over time, whether due to virtual machine migration or to physical mobility. A wide variety of methods are used to deal with these problems, ranging from sophisticated application layer technologies (i.e. NATS and load balancers) to awkward non-solutions (such as restarting the connection whenever a party to a new IP address).

Serval is one attempt to design a network architecture

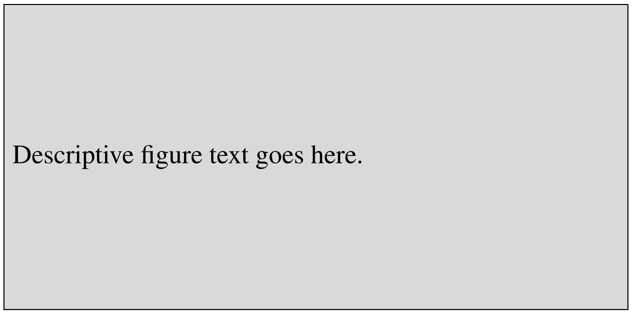

Figure 1: A placeholder caption for a placeholder figure.

more in line with the needs of today's Internet. As described in [6], Serval replaces TCP/IP's host-centric abstraction with a new service-centric abstraction. A connection is a link between two Service IDs, composed of one or more flows, each of which is a point-to-point link between two IP addresses. Instead of early-binding to an IP address using DNS, Serval-enabled hosts late-bind on Service IDs, which are mapped to the IP address of a target machine using anycast routing. Once a connection is established, either host can add additional flows or migrate existing flows without disrupting the ongoing connection, enabling multipath communication and seamless migration. All of this is made possible by the creation of a new Service Access Layer (SAL), sitting between the Network and Transport Layers. The SAL handles connection establishment and termination, flow creation, deletion, and migration, and

- Here

We have some related work in Section 5.

3 Design

4 Evaluation

5 Related Work

I like puppies [6][10][2][14][11][8][12][13][4][7][1][3][9].

6 Conclusion

Therefore, a duck.

References

[1] Allman, M., Paxson, V., and Blanton, E. TCP congestion control. RFC 5681 (Draft Standard), Sept. 2009.

[2] Arye, M., Nordstrom, E., Kiefer, R., Rexford, J., and Freedman, M. J. A formally-verified migration protocol for mobile, multi-homed hosts. In *Network Protocols (ICNP), 2012 20th IEEE International Conference on* (2012), IEEE, pp. 1–12.

[3] Ford, A., Raiciu, C., Handley, M., Barre, S., and Iyengar, J. Architectural guidelines for Multipath TCP development. RFC 6182 (Informational), Mar. 2011.

[4] Ford, B., and Iyengar, J. Breaking up the transport logjam. In *ACM HotNets, October* (2008).

[5] Gabrel, R. The rise of "worse is better". http://www.jwz.org/doc/worse-is-better.html, 1989. Accessed: 2013-05-06.

[6] Nordstrom, E., Shue, D., Gopalan, P., Kiefer, R., Arye, M., Ko, S. Y., Rexford, J., and Freedman, M. J. Serval: An end-host stack for service-centric networking. *Proc. 9th USENIX NSDI* (2012).

[7] Podmayersky, B. An incremental deployment strategy for Serval. Tech. Rep. TR-903-11, Princeton University Department of Computer Science, May 2011.

[8] Raiciu, C., Handley, M., and Wischik, D. Coupled congestion control for multipath transport protocols. *draft-ietf-mptcp-congestion-01 (work in progress)* (2011).

[9] Raiciu, C., Handley, M., and Wischik, D. Coupled congestion control for multipath transport protocols. RFC 6356 (Experimental), Oct. 2011.

[10] Raiciu, C., Paasch, C., Barre, S., Ford, A., Honda, M., Duchene, F., Bonaventure, O., and Handley, M. How hard can it be? Designing and implementing a deployable Multipath TCP. In *NSDI* (2012), vol. 12, pp. 29–29.

[11] Raiciu, C., Wischik, D., and Handley, M. Practical congestion control for multipath transport protocols. *University College London, London/United Kingdom, Tech. Rep* (2009).

[12] Wischik, D., Handley, M., and Braun, M. B. The resource pooling principle. *ACM SIGCOMM Computer Communication Review 38*, 5 (2008), 47–52.

[13] Wischik, D., Handley, M., and Raiciu, C. Control of Multipath TCP and optimization of multipath routing in the Internet. In *Network Control and Optimization*. Springer, 2009, pp. 204–218.

[14] Wischik, D., Raiciu, C., Greenhalgh, A., and Handley, M. Design, implementation and evaluation of congestion control for Multipath TCP. In *Proceedings of the 8th USENIX conference on Networked systems design and implementation* (2011), USENIX Association, pp. 8–8.

This paper represents my own work in accordance with University regulations.