R2D2: Bufferless, Switchless Data Center Networks Using Commodity Ethernet Hardware

Matthew P. Grosvenor Malte Schwarzkopf Andrew W. Moore
University of Cambridge Computer Laboratory
first.last@cl.cam.ac.uk

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: [Network topology]; C.2.5 [**Local and Wide-Area Networks**]: [Ethernet]

General Terms

Design, Performance

Keywords

Data centers, Latency, Scheduling, Broadcast, Ethernet

1. INTRODUCTION

Modern data centers commonly run distributed applications that require low-latency communication, and whose performance is critical to service revenue [1, 11]. If as little as one machine in 10,000 is a latency outlier, around 18% of requests will experience high latency [8].

Latency can originate at various sources in the software and hardware stack. In this work, we focus on in-network latency, which is an architectural network property and thus notoriously hard to improve *post-hoc* [6].

Ideally, in-network latency would be governed by the speed of light. In practice, however, this is not the case. This work seeks to answer why it is not. How close to the physical speed limit can we get, and what bounds can we guarantee?

2. NETWORK COSTS

Network latencies are composed of three components: *(i)* the time taken to serialize a packet onto the wire, *(ii)* the propagation delay once on wire, and *(iii)* in-network packet switching delays. Packet serialization is a function of the network bitrate and propagation delay is a function of the physical size of the network. Both are typically fixed in a data center network. Packet switching costs, however, are a consequence of the network architecture as well as traffic patterns experienced.

SIGCOMM'13, August 12–16, 2013, Hong Kong, China.
ACM 978-1-4503-2056-6/13/08.

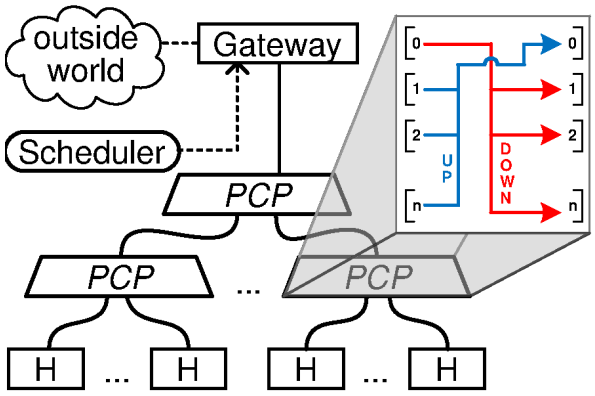

Figure 1: High-level R2D2 LLNet architecture: hosts (H) connected by PCPs to the gateway. Latency is guaranteed on the solid links.

Delays imposed by queuing in layer 2 switches are a significant contributor to in-network latencies in data centers, and thus recent work focused on keeping switch queue lengths to a minimum [1–3, 11]. Queuing should only occur when the ingress rate at a switch exceeds the available egress rate on a target port. TCP incast throughput collapse [5] is a common example. However, our experiments show that significant latency tails occur in switches merging low-rate flows (4Gb/s and 500Mb/s) even when sufficient egress bandwidth capacity is available. Our fast cut-through switch [4] with a measured best-case latency of under 800ns can degrade to tail-latencies of 1.1ms or more under these seemly innocuous conditions.

Unlike previous work, which attempts to circumvent switching costs by modifying the transport-layer protocol [1, 11] or explicit flow prioritization [2, 10, 12], we target the root of the problem and eliminate switch buffers completely. This approach is similar to that of pFabric [2], but more radical. We abandon the idea of a statistically multiplexed network in favour of an explicitly scheduled, contention-free, broadcast network. By seeking to eliminate in-network buffering, we demonstrate that that overall latency can be reduced and that strict, microsecond-level latency bounds can be enforced.

3. R2D2

The *Resilient Realtime Data Distributor* (R2D2), is a conceptually bufferless, switchless network architecture for datacenter "pods" of around 1,000 hosts. The core assumption

of R2D2 is that latency and bandwidth are separate concerns and should be treated accordingly. Therefore, we separate the network into two logical subnetworks: a contention-free, low-latency, broadcast network (*LLNet*), and a (potentially) deeply buffered, bandwidth optimised network (*BBNet*).

4. LLNET

Figure 1 shows the high-level architecture of an R2D2 LLNet. A *gateway* device marshals and schedules traffic atop a tree topology, connecting to end hosts at the leaves. It also connects to an external network ("outside world") that traffic may enter from or depart to.

The LLNet is supported by a new multi-port repeater device called a *passive cross-point* (PCP), which provides 1-to-N and N-to-1 connectivity without buffering and replaces network switches in the standard network (cabling) hierarchy. PCPs contain no active switching element. They are, therefore, the simplest possible multiplexing device, architecturally incurring near zero latency. In the downstream direction, the PCP acts like a hub, *i.e.* all downstream packets are transmitted broadcast. In the upstream direction, packets are merged without checking for collisions. Responsibility for scheduling packets through the PCP is shifted to the network scheduler, while the responsibility for queueing packets prior to entry is shifted to end-hosts' network adapters.

Ideally, PCPs would be realised entirely optically. In our work we have built an Optical-Electrical-Optical prototype PCP using the NetFPGA 10G card. In either case, PCPs are simpler and consume less energy than switches. Due to their simplicity, PCPs are less prone to failure and scale trivially with increasing bitrates.

Access to the LLNet is gated by a network scheduler. The scheduler sends specially crafted notification packets to ends hosts thereby allowing or preventing access to the network. This requires agreement between host NICs and the central scheduler on the meaning of notifications—a reasonable assumption in a datacenter under a single authority. Scheduling policy is application-specific: time-division multiplexing is the simplest policy, but we have also implemented several others.

Our prototype LLNet implementation achieves latencies of 35μs and 75μs in the 99.995%ile and 99.999%ile, respectively, for 1514-byte packets on a fully-loaded network. This figure includes crossing an unoptimized software implementation of gateway and scheduler, and traversing two PCPs.

5. BBNET

Conceptually, the LLNet is a shared medium where the overall bandwidth is limited to that of a single link. To improve bandwidth for bulk transfers, R2D2 also provides the BBNet. In its simplest guise, this is realized as a traditional, switched Ethernet topology. Other architectures, such as hypercube-based topologies [7, 9], are also possible and likely beneficial when optimizing for bandwidth.

6. R2D2 ON COMMODITY HARDWARE

While the two-network model is compelling, the practicalities of maintaining two physical networks may be unappealing. Since PCP devices embody a subset of the functionality found in network switches, it is possible to instead use the PCP as a conceptual model, realized on a commodity switch. Using a commercial cut-through switch can improve the performance of our LLNet to around 28μs. Since both the LLNet and BBNet can be implemented using the same commodity hardware, it is possible to merge these conceptually separate networks onto the same physical infrastructure, employing Ethernet's quality of service provisions to maintain isolation. R2D2 may therefore be realised on a single, unmodified, commodity Ethernet network, while still providing many of the benefits described above.

7. CONCLUSIONS

R2D2 is a work in progress. Substantial infrastructure construction and testing has already been completed and the prototype network is functional. We expect shortly to be able to implement a cluster-wide coherent memory cache that is unaffected by background traffic as a demonstration of its usefulness. We are also planning to investigate further scheduling policies and the latency guarantees they can provide.

8. ACKNOWLEDGEMENTS

This work was jointly supported by the EPSRC INTERNET Project EP/H040536/1 and the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-11-C-0249. The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

References

[1] ALIZADEH, M., ET AL. Data center TCP (DCTCP). In *Proceedings of SIGCOMM* (2010), pp. 63–74.

[2] ALIZADEH, M., ET AL. Deconstructing datacenter packet transport. In *Proceedings of HotNets* (2012), pp. 133–138.

[3] ALIZADEH, M., ET AL. Less is more: trading a little bandwidth for ultra-low latency in the data center. In *Proceedings of NSDI* (2012).

[4] ARISTA NETWORKS. 7150 Series 10G data sheet. `http://bit.ly/14rK1Nq`.

[5] CHEN, Y., ET AL. Understanding TCP incast throughput collapse in datacenter networks. In *Proceedings of WREN* (2009), pp. 73–82.

[6] CHESHIRE, S. It's the Latency, Stupid. `http://rescomp.stanford.edu/~cheshire/rants/Latency.html`; accessed 14/01/2013.

[7] COSTA, P., ET AL. CamCube: A Key-based Data Center. Tech. rep., MSR TR-2010-74, Microsoft Research, 2010.

[8] DEAN, J. AND BARROSO, L. The Tail at Scale: Managing Latency Variability in Large-Scale Online Services. *Communications of the ACM* (feb 2013).

[9] GUO, C., ET AL. Bcube: a high performance, server-centric network architecture for modular data centers. In *Proceedings of SIGCOMM* (2009), pp. 63–74.

[10] HONG, C., ET AL. Finishing flows quickly with preemptive scheduling. In *Proceedings of SIGCOMM* (2012), pp. 127–138.

[11] VAMANAN, B., ET AL. Deadline-aware datacenter tcp (D^{2}TCP). *SIGCOMM CCR* (Aug. 2012), 115–126.

[12] WILSON, C., ET AL. Better never than late: meeting deadlines in datacenter networks. In *Proceedings of SIGCOMM* (2011), pp. 50–61.