LiquidSwitch: Transport Enforcement for Datacenter Networks*

Keqiang He†, Eric Rozner‡, Kanak Agarwal‡, Yu Gu‡, Wes Felter‡, John Carter‡, Aditya Akella†
†UW-Madison ‡IBM
{keqhe,akella}@cs.wisc.edu {erozner,kba,yugu,wmf,retrac}@us.ibm.com

1. INTRODUCTION

In multi-tenant datacenters, VMs play an integral role by enabling a diverse set of operating systems and software to be run on a unified underlying framework. However, out-dated, inefficient, or misconfigured TCP stacks can be implemented in the VMs. In this paper, we explore how operators can regain control of TCP's congestion control, regardless of the TCP stack running in a VM. Our aim is to allow a cloud provider to utilize advanced TCP stacks, such as DCTCP [1], without having control over the VM or requiring changes in network hardware. We propose a scheme that exerts fine-grained control over arbitrary tenant TCP stacks by enforcing per-flow congestion control in the virtual switch.

2. OUR APPROACH

We present LiquidSwitch, a new technology that implements TCP congestion control within a vSwitch to help ensure VM TCP performance cannot impact the network in an adverse way. At a high-level (Figure 1), the vSwitch monitors all packets for a flow, modifies packets to support features not implemented in the VM's TCP stack (*e.g.*, ECN) and reconstructs important TCP parameters for congestion control. LiquidSwitch runs the congestion control logic specified by an administrator and then enforces an intended congestion window by modifying the receiver advertised window (`RWND`) on incoming ACKs. A policing mechanism ensures stacks cannot benefit from ignoring `RWND` and can also be used for non-TCP traffic.

LiquidSwitch allows for administrators to enforce a uniform, network-wide congestion control algorithm without changing VMs. DCTCP congestion control algorithm is implemented in LiquidSwitch, this allows for high throughput and low latency, regardless of the congestion control algorithms VMs use. Furthermore, our system mitigates the impact of varying TCP stacks running on the same fabric. This improves fairness and additionally solves the ECN co-existence problem identified in production networks [2, 3]. LiquidSwitch is easy to implement, computationally lightweight, scalable and modular.

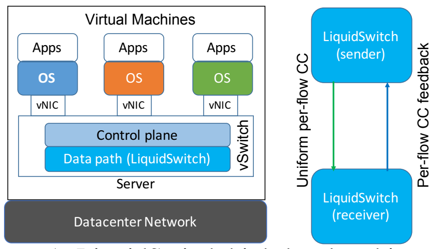
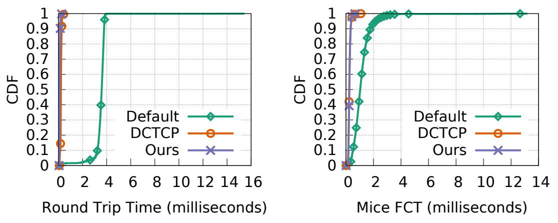


Figure 1: LiquidSwitch high-level architecture.



(a) TCP RTT in 16-to-1 incast. (b) mice flow (16KB) FCT.
Figure 2: LiquidSwitch Performance

3. EXPERIMENT RESULTS

We attach 17 servers to a IBM G8264 10Gbps switch. We measure Default (CUBIC stack without ECN) and DCTCP (ECN-enabled) on an unmodified vSwitch. We compare them to LiquidSwitch (CUBIC on host with LiquidSwitch-based vSwitch). Our evaluation (Figure 2) shows LiquidSwitch closely tracks DCTCP's performance regardless of the VM's TCP stack and outperforms Default. The computational overhead of LiquidSwitch is less than 4% compared with the baseline case.

References

[1] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). In *SIGCOMM*, 2010.
[2] G. Judd. Attaining the Promise and Avoiding the Pitfalls of TCP in the Datacenter. In *NSDI*, 2015.
[3] H. Wu, J. Ju, G. Lu, C. Guo, Y. Xiong, and Y. Zhang. Tuning ECN for Data Center Networks. In *CoNEXT*, 2012.

*Keqiang He is a student author