WVCC: Weighted Virtual Congestion Control for Datacenter Networks

Jiaqing Dong
Department of Computer Science
Tsinghua University

Yi Wang*
Future Network Theory Lab
Huawei

Chen Tian
State Key Laboratory for Novel Software Technology
Nanjing University

Bo Jin
Future Network Theory Lab
Huawei

Hao Yin
Department of Computer Science
Tsinghua University

Gong Zhang
Future Network Theory Lab
Huawei

ABSTRACT

Enforcing virtualized congestion control is a new trend for datacenter networks. Virtual Congestion Control (VCC) can also enforce differentiated Quality-of-Service (QoS) for flows. However, current flow differentiation algorithms only provide qualitative rather than quantitative bandwidth allocation. Weighted bandwidth allocation is critical to enforce administrator policy. In this work, we propose Weighted Virtual Congestion Control (WVCC) enforcement for datacenter networks. It is a novel per-flow differentiation mechanism capable of proportionally allocating bandwidth among flows.

CCS CONCEPTS

• **Networks** → **Transport protocols**; *Data center networks*;

KEYWORDS

Network Proportionality, Congestion Control, Virtualization

ACM Reference format:
Jiaqing Dong, Yi Wang, Chen Tian, Bo Jin, Hao Yin, and Gong Zhang. 2017. WVCC: Weighted Virtual Congestion Control for Datacenter Networks. In *Proceedings of SIGCOMM Posters and Demos '17, Los Angeles, CA, USA, August 22–24, 2017,* 3 pages.
https://doi.org/10.1145/3123878.3131993

1 PROBLEM STATEMENT

Enforcing virtualized congestion control is a new trend for datacenter networks [2, 3]. Public cloud datacenters are shared by various tenants running Virtual Machines (VM). Guest VMs have different TCP protocol stacks. These TCP versions could rely on different congestion signals (*e.g.*, ECN vs. packet drop) and exert different control laws. Some of these mechanism might be even out-dated. Nevertheless, they cannot peacefully share the same underlying physical datacenter network [5]. AC/DC [3] and vCC [2] add a translation layer which "hijacks" the congestion control function in the datapath. This *Virtual Congestion Control* (VCC) mechanism translates the legacy TCP versions in tenant VMs into a newer congestion control algorithm. As a result, it allows guest-VM applications continue to use their legacy TCP implementations. At the same time, the administrator can enforce a uniform congestion control rule throughout the whole datacenter.

[*] Yi Wang is the corresponding author.

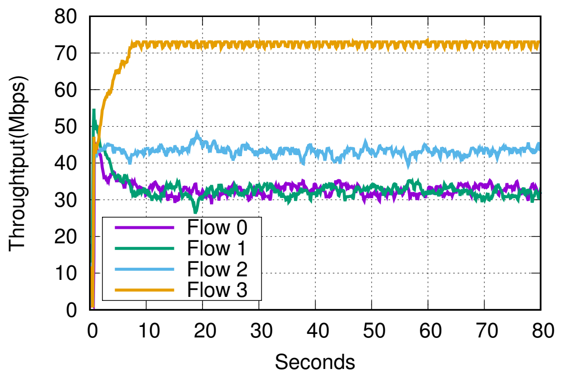


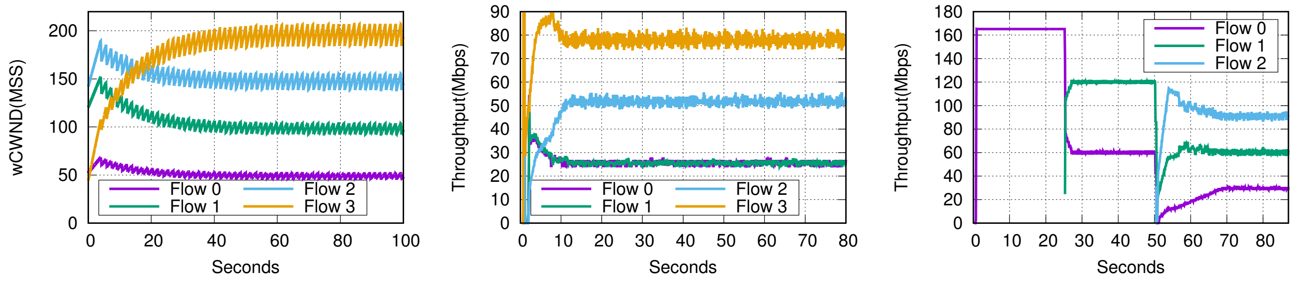
Figure 1: Per-flow differentiation of AC/DC.

VCC can also enforce differentiated Quality-of-Service (QoS) for flows. AC/DC emulates a DCTCP-like [1] congestion control algorithm at host virtual switch. On top of that, AC/DC proposes a per-flow differentiation mechanism by assigning each flow with a priority β and changes the back-off phase of DCTCP as:

$$rwnd \leftarrow rwnd \times (1 - \alpha(1 - \frac{\beta}{2})). \qquad (1)$$

With Equation 1, flows with lower priority back-off more aggressively than higher-priority flows.

The per-flow differentiation algorithm based on Equation 1 only provides qualitative rather than quantitative bandwidth allocation among flows. We use an NS3 simulation to illustrate this problem, where four flows with β values $[1, 1, 2, 3]/4.0$ competing at a single link. As shown in Figure 1, after convergence, flows with the same priority get similar throughput. While flows with higher priority obtain higher throughput. However, priority value β cannot provide proportional bandwidth allocation among these flows quantitatively. The reason is that β in Equation 1 is a priority value which only describes the back-off strength, without considering the increase phase. The priority parameter β cannot be translated into weight due to failing to consider the increase phase. Detailed analysis will be available in our future work.



(a) Results of the fluid model

(b) Four flows in WVCC

(c) Flows arriving at intervals in WVCC

Figure 2: Evaluation results of WVCC.

Weighted bandwidth allocation is critical to enforce administrator policy [5]. Seawall [4] provides proportional bandwidth allocation among entities by forcing traffic through congestion-controlled tunnels configured with weights. However, it cannot not support bandwidth allocation at flow-level. Software rate limiters like token bucket filters (TBF) cannot scale to flow-level granularity in datacenter scenarios.

In this work we propose Weighted Virtual Congestion Control (WVCC) enforcement for datacenter networks. It is a novel per-flow differentiation mechanism capable of proportionally allocating bandwidth among flows. Inherited from VCC, WVCC does not require any modifications to legacy TCP stacks in tenant-side VMs.

2 WEIGHTED VCC

Overview: WVCC is implemented in the datapath of the hypervisor, similar to AC/DC [3] and vCC [2]. With this architecture, WVCC can take effect without any TCP stack modifications in tenant-side environments. The sender and receiver modules together enforce the per-flow weighted congestion control algorithms by changing the receive window (RWND) field in incoming ACK packets. Specifically, a WVCC algorithm calculates the weighted congestion window ($cwnd^*$) value. RWND is modified only when $cwnd^*$ is smaller than the initial RWND set by the receiver.

Algorithm: WVCC algorithm builds on top of DCTCP [1]. Switches are required to mark CE bits when packets in buffer exceeds a destined threshold. Like DCTCP, we maintain the same variable α, which can be used to quantitatively measure the extent of congestion in the network. WVCC shares same features with TCP, such as slow start, congestion avoidance, and fast recovery. Upon receiving an ACK, WVCC calculates new congestion window[1]:

$$cwnd^* \leftarrow \begin{cases} cwnd^* \times (1-\alpha/2), & \text{ECE bit of ACK set;} \\ cwnd^* + w/cwnd^*, & \text{otherwise.} \end{cases} \quad (2)$$

where w ($w < 1$) is the weight given to that flow. In implementation, equation 2 alters into a much more simplified version:

$$cwnd^* \leftarrow w \times cwnd, \quad (3)$$

where $cwnd$ is the congestion window calculated by DCTCP stack upon each ACK. After all the calculations in the stack, WVCC calculates a weighted congestion window $cwnd^*$ as equation 3 and leaves $cwnd$ unchanged in the stack. As $cwnd$ remains untouched in the stack, no modifications are required to the original DCTCP behaviors and equation 3 naturally satisfies equation 2.

Analysis: We develop a fluid model which describes how WVCC calculates wCWNDs for flows with different weights. Due to space limitation, we omit the details.

[1]Window cut happens at most once per window of data

3 EVALUATION

Fluid Model: We implement the fluid model in Matlab. In the emulation, four flows are setup with weights from 1 to 4. Each flow starts with a random initial congestion window. In Figure 2(a), it is observed that the four flows converge to the steady state with wCWND sizes proportional to their weights.

NS3 Simulation: We implement WVCC in NS3 and demonstrate the flow-level proportional bandwidth allocation capability. To compare with results of AC/DC in Figure 1, we setup four flows weighted $[1, 1, 2, 3]/4.0$, starting concurrently. In Figure 2(b), these four flows converges quickly and each of them achieves the throughput quantitatively proportional to their weights.

Furthermore, to demonstrate the correctness of WVCC in dynamic scenarios, we setup three flows in different hosts with weights $[1, 2, 3]/4.0$, competing on the same bottleneck link and starting with intervals. Figure 2(c) shows that every time a new flow arrives, all flows converge to a new steady state with bandwidth sharing proportional to their weights.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable comments. This work was supported in part by the National Key Research and Development Program under Grant no. 2016YFB1000102, the National Science and Technology Major Project of China under Grant Number 2017ZX03001013-003, the Fundamental Research Funds for the Central Universities under Grant Number 0202-14380037, the National Natural Science Foundation of China under Grant no. 61672318, 61631013, 61602194, 61402198, 61321491, and the projects of Tsinghua National Laboratory for Information Science and Technology (TNList).

REFERENCES

[1] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. 2010. Data Center TCP (DCTCP). In *Proceedings of the ACM SIGCOMM 2010 Conference (SIGCOMM '10)*. ACM, New York, NY, USA, 63–74. https://doi.org/10.1145/1851182.1851192

[2] Bryce Cronkite-Ratcliff, Aran Bergman, Shay Vargaftik, Madhusudhan Ravi, Nick McKeown, Ittai Abraham, and Isaac Keslassy. 2016. Virtualized Congestion Control. In *Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM '16)*. ACM, New York, NY, USA, 230–243. https://doi.org/10.1145/2934872.2934889

[3] Keqiang He, Eric Rozner, Kanak Agarwal, Yu (Jason) Gu, Wes Felter, John Carter, and Aditya Akella. 2016. AC/DC TCP: Virtual Congestion Control Enforcement for Datacenter Networks. In *Proceedings of the 2016 ACM SIGCOMM Conference (SIGCOMM '16)*. ACM, New York, NY, USA, 244–257. https://doi.org/10.1145/2934872.2934903

[4] Alan Shieh, Srikanth Kandula, Albert Greenberg, and Changhoon Kim. 2010. Seawall: Performance Isolation for Cloud Datacenter Networks. In *Proceedings of the 2Nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud'10)*. USENIX Association, Berkeley, CA, USA, 1–1. http://dl.acm.org/citation.cfm?id=1863103.1863104

[5] Chen Tian, Ali Munir, Alex X Liu, Yingtong Liu, Yanzhao Li, Jiajun Sun, Fan Zhang, and Gong Zhang. 2017. Multi-Tenant Multi-Objective Bandwidth Allocation in Datacenters Using Stacked Congestion Control. In *INFOCOM, 2017 Proceedings IEEE*. IEEE, 3074–3082.