Case Study of Deadlock Problem in RDMA Datacenter Networks

June 2, 2016 4:28 AM

Abstract

to be added.

1 Introduction

Driven by the need for ultra-low latency, high throughput and low CPU overhead, both Microsoft and Google are deploying Remote Direct Memory Access (RDMA) in their datacenter networks in recent years [7, 8]. Among the available RDMA technologies, RDMA over Converged Ethernet (RoCE) [1] is a promising one as it is compatible with current IP and Ethernet based datacenter networks.

The deployment of RoCE requires Priority-based Flow Control (PFC) [2] to provide a lossless L2 network. With PFC, packet loss can be avoided by letting a switch pause its immediate upstream device before buffer overflow occurs. However, the adoption of PFC will cause deadlock problem. Deadlock refers to such a standstill situation: There is a cyclic buffer dependency among a set of switches. Any switch in the cycle holds all the buffer needed by its upstream switch, and meanwhile is waiting for its downstream switch to release some buffer and resume its packet transmission.

It is easy to see that when deadlock occurs, no switch in the cycle can proceed. Further, throughput of the whole network or part of the network will go to zero due to the backpressure effect of PFC pause. Hence it is necessary to design some mechanisms for handling deadlock problem when deploying RDMA in datacenter networks.

Prior works on preventing deadlock can be roughly classified into two categories including 1) *Routing restriction based approach* [3, 6]. The idea of this approach is to ensure that no cyclic buffer dependency exists in the network by limiting the routing paths used in each priority class; 2) *buffer management (structured buffer pool) based approach* [4, 5]. This approach divides switch buffer into several buffer classes. A packet is allowed to access more buffer classes as it travels greater distance in the network. It can be proved that as long as the number of buffer classes is no smaller than the hop count of the longest routing path, there will be no cyclic buffer dependency in the network.

These two kinds of approaches are known to have some important drawbacks. For example, routing restriction based approach usually wastes link bandwidth and limits throughput performance, while buffer management based approach introduces non-trivial deployment complexity to the network system. The drawbacks of these approaches somehow can be viewed as the cost to eliminate cyclic buffer dependency in the network, which has been identified as the primary principle for designing a deadlock prevention solution.

Though obeying this principle can guarantee a deadlock-free network, its cost can be very expensive. In this paper, instead of seeking a solution that introduces less overhead, we take one step back and ask: Is cyclic buffer dependency a sufficient and necessary condition for deadlock? Or can deadlock-free be guaranteed without eliminating cyclic buffer dependency?

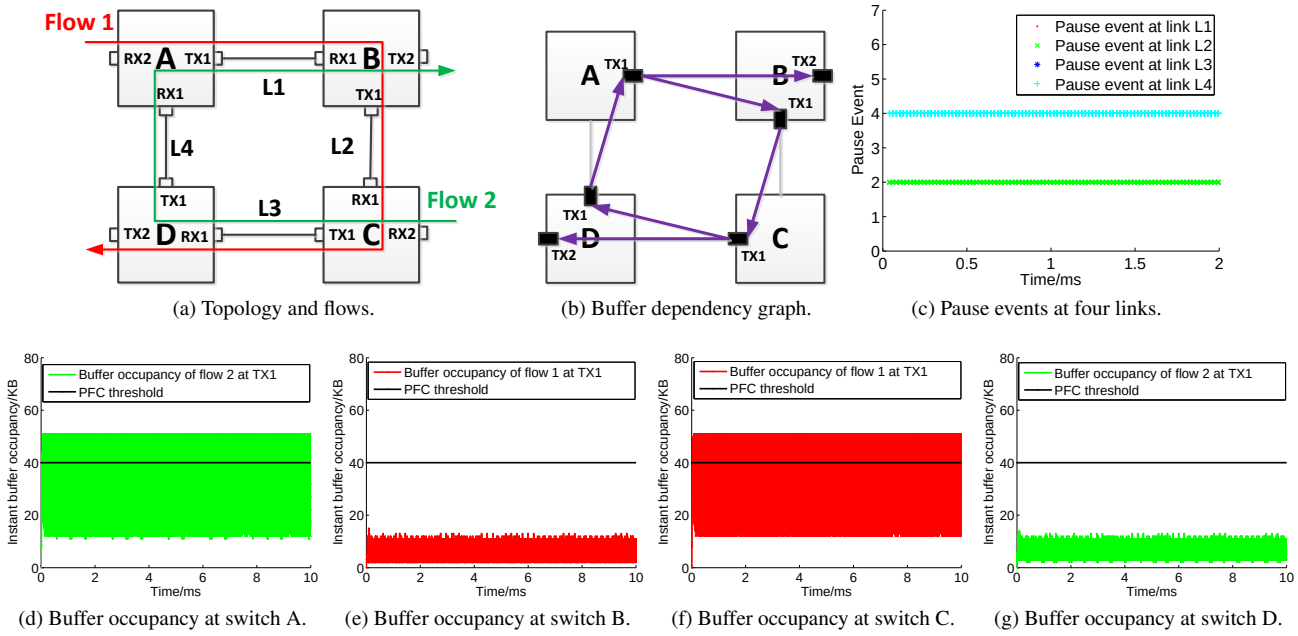
To answer the above questions, we did some study about several representative deadlock cases. Our findings are as follows. First, cyclic buffer dependency is just a necessary condition for deadlock. There are some cases where cyclic buffer dependency is met, but there is no deadlock. Second, even if all the links in a switch cycle are paused simultaneously, deadlock may still not occur. These two findings indicate that prior solutions are designed based on a too strict condition, and thus introduces some unnecessary overhead. (We need a much better discussion here, let's discuss how to argue that our deadlock case study is important and meaningful.)

2 Deadlock case study

In this part, we present our simulation study about three deadlock cases, and demonstrate that 1) cyclic buffer dependency is not a sufficient and necessary condition for deadlock; 2) simultaneous cyclic pause is not sufficient to create a permanent deadlock.

Simulation setup: To create a well controlled experimental environment for deadlock analysis, we did our deadlock case study using packet-level NS-3 simulations.

In our modified NS-3 simulator, we implement PFC protocol (i.e., IEEE 802.1 Qbb protocol). Note that most



(a) Topology and flows. (b) Buffer dependency graph. (c) Pause events at four links.

(d) Buffer occupancy at switch A. (e) Buffer occupancy at switch B. (f) Buffer occupancy at switch C. (g) Buffer occupancy at switch D.

Figure 1: Deadlock case 1.

modern commodity switches are output-queued, while PFC works in a per ingress queue fashion. Basically, for each ingress queue, the switch will maintain a counter to track its instant virtual queue length (i.e., bytes of buffered packets received by this ingress queue). Once the queue length of an ingress queue exceeds the pre-configured PFC threshold, a pause frame will be sent to the corresponding upstream device. The upstream device then stop sending any packet to this ingress queue unless 1) the pause frame has expired; 2) or it has received a resume frame from this ingress queue.

In our simulations, link capacity of all links is 40Gbps. All the switches have 12MB buffer. PFC threshold is statically set to 40KB for each ingress queue.

Case 1: In this case, as shown in Fig. 1(a), we let two flows run over four switches A, B, C and D. Flow 1 starts at a host (not shown) attached to A, passes through B and C, and ends at a host attached to D. Flow 2 starts at a host attached to C, passes through D and A, and ends at a host attached to B. In the figure, RX represents input queue (or port), while TX represents output queue (or port). To evaluate whether there will be deadlock in the worst case, both flows are UDP flows with infinite traffic demand.

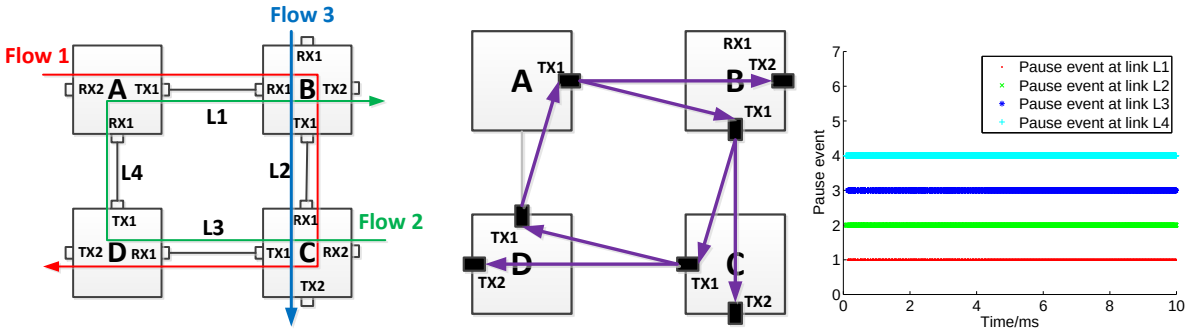
Buffer dependency graph of case 1 is drawn in Fig. 1(b). Each directed line represents a buffer dependency from the source TX to the destination TX. For example, packets buffered in TX1 of A will be sent to either TX1 or TX2 of B, so in Fig. 1(b), two directed lines are drawn there between A and B. Similarly, we can draw the dependency lines between other switches. As we can see in Fig. 1(b), there is a cyclic buffer dependency among the four switches, i.e., dependencies from TX1 of A to TX1 of B, then to TX1 of C, then to TX1 of D, and finally back to TX1 of A.

In Fig. 1(c), we plot the PFC pause events at four links L1, L2, L3 and L4. Basically, if link Li (i=1,2,3,4) is paused at time t, we plot a point at location (t, i). Pause events at different links are plotted with different colors and of different heights. As we can observe, links L2 and L4 are paused continuously, while the other two links L1 and L3 never get paused. In this case, deadlock will never occur as no packet will be paused permanently.

To understand the pause pattern in Fig. 1(c), we sample the instant buffer occupancy of both flows at TX1 queues of A, B, C and D every 1us. In Fig. 1(d), we draw the instant buffer occupancy of flow 2 at TX1 of A. Buffer occupancy of flow 1 is not drawn in Fig. 1(d) as it does not contribute to the pause of link L1 (Note that PFC works in a per ingress queue fashion).

Similarly, in Fig. 1(e), Fig. 1(f) and Fig. 1(g), we draw the instant buffer occupancy of interested flows at TX1 queues of B, C and D, respectively. As flow 1 and flow 2 are symmetric, we only present the analysis for Fig. 1(d) and Fig. 1(e) to show why Link L4 is paused continuously but link L1 never gets paused.

As shown in Fig. 1(d), buffer occupancy of flow 2 at TX1 of A fluctuates between 10KB and 55KB around the PFC threshold, so link L4 will get paused intermittently. In contrast, buffer occupancy of flow 1 at TX1 of B is well below the PFC threshold (fluctuates between 0KB



(a) Topology and flows. (b) Buffer dependency graph. (c) Pause events at four links.

Figure 2: Deadlock case 2.

and 18KB), hence link L1 will never be paused.

Observation 1: Deadlock may not occur when cyclic buffer dependency exists in the network. Cyclic buffer dependency is just a necessary condition for deadlock.

Case 2: as shown in Fig. 2(a), in the second case, in addition to case 1, we add another flow (flow 3) to run over switches B and C in sequence. All the three flows are UDP flows with infinite traffic demand. Buffer dependency graph of case 2 is drawn in Fig. 2(b). Compared with case 1, one additional dependency from TX1 of B to TX1 of C is added.

Pause events at four links L1, L2, L3 and L4 are plotted In Fig. 2(c). As we can see, in this case four links are all paused. To check whether deadlock will occur in this case, we stop the three flows after a sufficient long period (1000ms). What we find is that, pause events are continuously generated at all the four links even after three flows stop sending new packets (not shown in the figure). This means that a deadlock has been created among switches A, B, C and D.

In the next, we will explain why deadlock can be created by adding one additional flow to the network. After adding flow 3, packets of flow 1 buffered in TX1 of B can no longer get transmitted at full link speed due to the contention with packets of flow 3. Packets of flow 1 will then build up at TX1 of B. Once the packets of flow 1 buffered at TX1 of B exceed the PFC threshold, RX1 of B will send a pause frame to TX1 of A to pause Link L1. The pause on link L1 will help packets to build up at TX1 of A, and has a cascade effect on link L4. Due to the pause at link L1, link L4 will get paused more frequently. Then packets of flow 2 are easier to build up at TX1 of D. Once the PFC threshold is triggered at D, link L3 also get paused. So in case 2, pause events can occur at all the four links.

Once all the four links are paused simultaneously, there is a chance that no link can get resumed. For example, it is possible that when simultaneous pause happens, at switch A and switch B, the first packet buffered in the head of TX1 is a packet of flow 1, and meanwhile, at switch C and switch D, the first packet buffered in the head of TX1 is a packet of flow 2. If the above condition occurs, no link can get resumed as all the TX1 queues are waiting for its downstream neighbors to release some buffer to break the standstill condition.

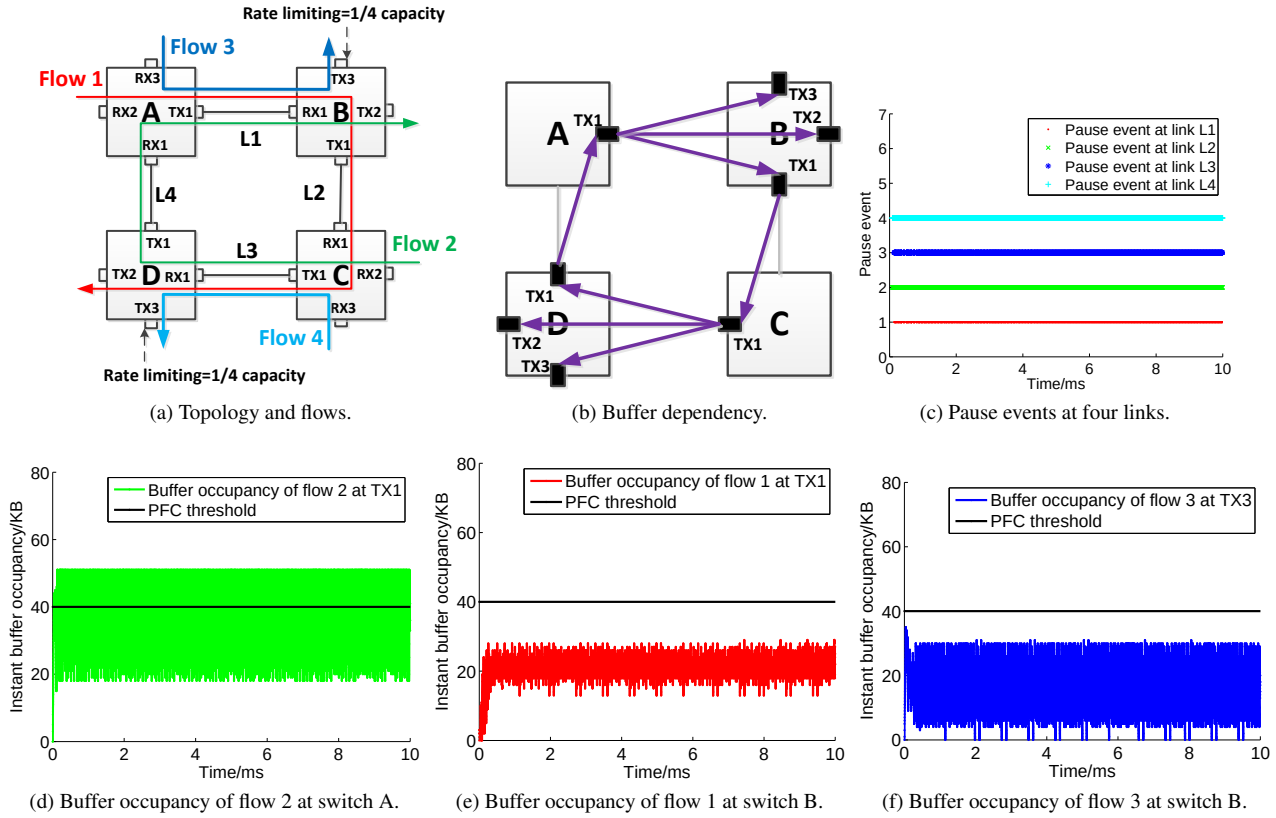
Observation 2: Deadlock will occur when all the links in a cycle are paused simultaneously and no link can get resumed.

Case 3: In this case, we will show that even if all the four links are paused simultaneously, it is not guaranteed that a deadlock will be created. As shown in Fig. 3(a), in addition to case 1, we add another two flows (flow 3 and flow 4). Flow 3 starts at a host attached to A, passes through A and B, and ends at a host attached to B. Flow 4 is a symmetric flow of flow 3 that runs over C and D. All the four flows are UDP flows with infinite traffic demand. Two rate limiters are added at TX3 queues of B and D to ensure that rates of flow 3 and flow 4 will not exceed $1/4$ of the link capacity. Buffer dependency graph of case 3 is drawn in Fig. 3(b). Compared with case 1, two additional buffer dependencies from TX1 queues to TX3 queues are added.

Pause events at four links L1, L2, L3 and L4 are plotted In Fig. 3(c). As we can see from the figure, all the four links are paused continuously. However, we find that once we stop the flows, all the links get resumed and buffer occupancy of four switches soon becomes zero. This indicates that deadlock cannot be created in case 3.

To find out why there is no deadlock in case 3, we draw the instant buffer occupancy of 3 flows at switches A and B in Fig. 3(d), Fig. 3(e) and Fig. 3(f). Here we omit the buffer occupancy condition of switches C and D as the topology and the flows are symmetric.

As shown in Fig. 3(d), at TX1 of A, the buffer occupancy of flow 2 exceeds the PFC threshold, so link L4 will get paused. To understand why link L1 is also paused, we need to consider the buffer occupancy of flow 1 and flow 3 at switch B. The reason is that packets re-



(a) Topology and flows.

(b) Buffer dependency.

(c) Pause events at four links.

(d) Buffer occupancy of flow 2 at switch A.

(e) Buffer occupancy of flow 1 at switch B.

(f) Buffer occupancy of flow 3 at switch B.

Figure 3: Deadlock case 3.

ceived by RX1 of B are possible to be queued at both TX1 and TX3 (note that there is a rate limiter on TX3). As long as the sum of the buffer occupancies of both TX queues exceeds the PFC threshold, link L1 will get paused. As we can see in Fig. 3(e) and Fig. 3(f), although individually buffer occupancy of either TX1 or TX3 is less than the PFC threshold, their sum is larger than the PFC threshold. Hence link L1 is paused.

As both TX1 and TX3 contribute to the pause on link L1, to create a deadlock, we need to ensure that packets buffered at both TX queues cannot get resumed. However, packets buffered at TX3 can always get transmitted within a finite time as it is not involved in any cyclic buffer dependency. This explains why when we stop the flows, all the four links can be resumed from the pauses.

Observation 3: even if all the links in a cycle are paused simultaneously, it is not sufficient to create a permanent deadlock.

References

[1] "RDMA over Converged Ethernet (RoCE)," http://www.mellanox.com/page/products_dyn?product_family=79.

[2] "IEEE. 802.11Qbb," ser. Priority based flow control, 2011.

[3] J. Flich, T. Skeie, A. Mejia, O. Lysne, P. Lopez, A. Robles, J. Duato, M. Koibuchi, T. Rokicki, and J. C. Sancho, "A survey and evaluation of topology-agnostic deterministic routing algorithms," *IEEE Transactions on Parallel and Distributed Systems*, 2012.

[4] M. Gerla and L. Kleinrock, "Flow control: A comparative survey," *IEEE Transactions on Communications*, 1980.

[5] M. Karol, S. J. Golestani, and D. Lee, "Prevention of deadlocks and livelocks in lossless backpressured packet networks," *IEEE/ACM Transactions on Networking*, 2003.

[6] R. Mittal, V. T. Lam, N. Dukkipati, E. Blem, H. Wassel, M. Ghobadi, A. Vahdat, Y. Wang, D. Wetherall, and D. Zats, "Brent, Stephens and Alan, L. Cox and Ankit, Singla and John, Carter and Colin, Dixon and Wesley, Felter," ser. INFOCOM '14.

[7] R. Mittal, V. T. Lam, N. Dukkipati, E. Blem, H. Wassel, M. Ghobadi, A. Vahdat, Y. Wang, D. Wetherall, and D. Zats, "TIMELY: RTT-based Congestion Control for the Datacenter," ser. SIGCOMM '15.

[8] Y. Zhu, H. Eran, D. Firestone, C. Guo, M. Lipshteyn, Y. Liron, J. Padhye, S. Raindel, M. H. Yahia, and M. Zhang, "Congestion Control for Large-Scale RDMA Deployments," ser. SIGCOMM '15.