Improving resilience of RDMA data centers

November 30, 2015 9:29 PM

Abstract

to be added.

1 Introduction

to be added.

2 Background

2.1 Routing loops in DCNs

Various loop incidents have been observed in both Google's and Microsoft's production DCNs [2, 4]. In DCNs, routing loops arise when there are inconsistencies in routing state among a set of switches. Routing inconsistencies can be caused by many network errors including corruption of forwarding entries, failure of devices and misconfiguration of network devices.

Routing loops can be either persistent or transient. Persistent loops are commonly caused by misconfiguration or device bugs, and require human intervention to fix them. Transient loops arise as different switches in a network usually converge at different speed when a network failure occurs. The duration of a transient loop is directly related to the convergence time of the routing protocol in use. The convergence times of different routing protocols, e.g., BGP [1], are different and are affected by the network scale.

2.2 Priority-based Flow Control in RDMA DCNs

Driven by the rising demand of cloud applications, we are deploying RoCEv2-based RDMA in Microsoft's DCNs to provide ultra-low latency (10us per hop or less) and high throughput (40Gbps or more), with very low CPU overhead. The deployment of RoCEv2-based RDMA needs Priority-based Flow Control (PFC) to enable a lossless Ethernet fabric [3]. When a switch port or a NIC is unable to accommodate more packets in its buffer, PFC can prevent buffer overflow by sending PAUSE frames to its directly connected upstream device to stop further data transmission.

In more details, switches and NICs track the queue length of each ingress queue[1]. Once the queue length exceeds a certain threshold, a PAUSE frame will be sent to the upstream device. The upstream device then stops transmission on that link for some amount of time specified in the PAUSE frame or until receive a RESUME frame from its downstream device.

PFC has a cascade effect: Once a link is paused for a long time due to traffic congestion, related interfaces of upstream devices will be paused hop-by-hop backward until traffic sources are finally paused. This cascade effect will not be a problem if PFC can operate at a per flow level. However, due the the hardware limitation, PFC now operates at a per port (or, port plus priority) level and can at most support 8 priority classes, which means that a flow is possible to be paused by a congested link that is not even on its path.

The using of PFC mainly introduces two problems: congestion spreading problem and deadlock problem.

Under normal circumstances (i.e., no network error happens), congestion spreading problem can be well handled by running an effective flow-level congestion control protocol at all server NICs, e.g., DCQCN [3] is such an protocol we have designed for our RoCEv2-based RDMA DCNs. But in practice, it is difficult to guarantee that the network will never run into any incorrect state, especially for large scale DCNs consists of tens of thousands of switches and millions of servers. We found that the behaviour of PFC is dangerous when the network is in an incorrect state. With PFC, even a transient routing loop is possible to bring the whole DCN into a permanent deadlock state. More details will be introduced in the next part.

3 Analysis of deadlock problem

3.1 Deadlock problem in RDMA DCNs

Once a loop occurs in a network, packets of some flows will be caught in the loop and traverse the same links

[1] Packets are actually only buffered in the egress queues, but the device will maintain a counter to track the queue length of each virtual ingress queue.

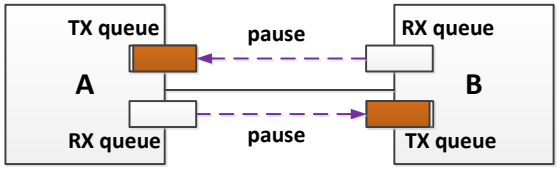

Figure 1: An example of loop induced deadlock: there is a loop between switch A and switch B. Both TX queues (egress queues) are paused by PFC as no buffer are available at both switches to accommodate more packets.

multiple times until they are dropped due to Time-to-Live (TTL) expiration. Apart from causing packet drops, loops will also waste some link bandwidth as well as increase the end-to-end delay for the flows traversing some link(s) in the loop (but not caught by the loop).

In a lossy network, the impact of a loop is not fatal and can be completely eliminated as long as the loop is removed from the network. In contrast, in a lossless network, if packets enter a loop faster than they get dropped in the loop due to TTL expiration, packets will occupy the buffer of all the switches in the loop, and then a deadlock is created. When a deadlock occurs, each switch in the loop is paused by its downstream switch, and at the same time pauses its upstream switch due to the lack of available buffer to accept more packets. Once such a circular buffer dependency is created, the deadlock condition will hold persistently even after the loop is eliminated.

Under deadlock condition, no packets can move along the links in the loop, and more and more devices outside the loop will be paused due to the cascade effect of PFC. If a deadlock is created in the core of the network, it is very likely to bring the whole data center into a deadlock state.

A simple deadlock example is shown in Fig. 1. In this example, there is a routing loop between switch A and switch B. Packets enter this loop at a sufficient large rate and soon occupy all the available buffer of both switches. Then Both TX queues (egress queues) will be paused by PFC PAUSE frames and a deadlock is created. As we can see, this deadlock cannot be resolved by eliminating the routing loop as packets are already queued in the TX queues and can never reach the next-hop switch to escape from the loop.

3.2 Sufficient condition for deadlock creation

In this part, we analyze the sufficient condition to create a deadlock when there is a loop in the network.

At first, we consider the maximum packet drain rate

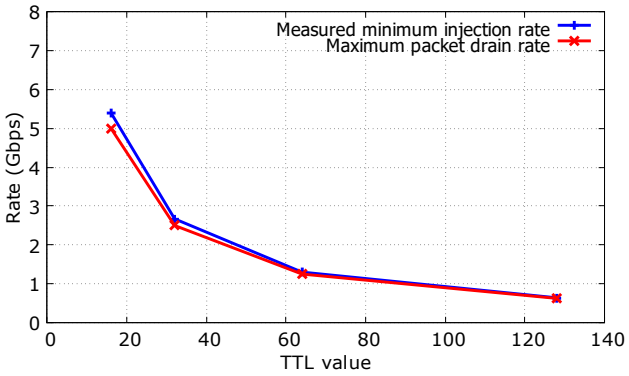

Figure 2: Measurement of the minimum injection rate to create a deadlock. Both switches in the loop are Arista 7050QX32.

in a loop regarding TTL expiration. Let n the number of switches in a loop, B be the link bandwidth and k_{TTL} be the TTL value of packets before they enter the loop. Each time a packet traverses one switch, its TTL value will be reduced by 1.

The maximum packet drain rate is achieved when no switch is paused by PFC PAUSE frame and each switch is sending packets to its next-hop in the loop at the rate of B. So the maximum packet drain rate r_d^{max} is equal to nB/k_{TTL}. here nB can be viewed as the maximum packet "flowing" rate in the loop, while $1/k_{TTL}$ captures the information that a packet will be dropped after it has traversed k_{TTL} hops of switches in the loop.

Let r_{in} be the injection rate of packets into the loop. One sufficient condition to create a deadlock in a lossless network is that: there is a loop in the network, and condition $r_{in} > r_d^{max}$ holds for a sufficient long period until a circular buffer dependency is created in the loop.

To verify our analysis above, we manually configure a loop between two 40Gbps switches, and measure the minimum packet injection rate that can create a deadlock. The result is shown in Fig. 2. As we can see, the measured minimum packet injection rate is just slightly larger than the maximum packet drain rate which is computed according to nB/k_{TTL}. This observation holds when TTL is set to different values.

Another observation from the figure is that, setting smaller TTL can help to prevent deadlock but its benefit is limited. As shown in the figure, when TTL is set to 16 which is already a very small value, the minimum injection rate is only about 6Gbps.

3.3 Analysis of the time to create a deadlock

In this part, we analyze and measure the time to create a deadlock when the sufficient condition for deadlock cre-

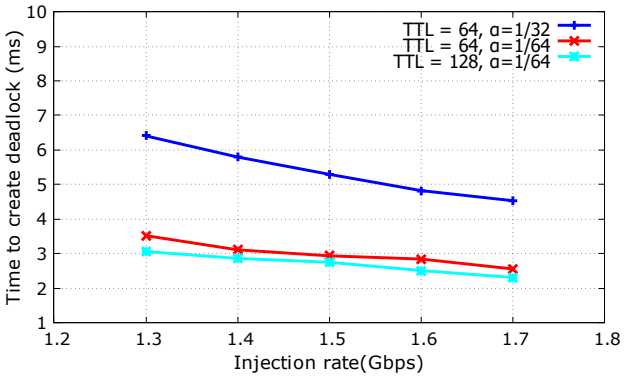

Figure 3: Measurement of the time to create a deadlock under different settings (deadlock will not occur when the injection rate is less than 1.3Gbps).

ation is already met. Deadlock creation time is related to three factors: **packet injection rate** r_{in}, **packet drain rate** r_d and **PFC threshold** t_{PFC}.

t_{PFC} determines the minimum bytes of packets needed to be "trapped" in the loop to create a deadlock, while $r_{in} - r_d$ can be viewed as the packet increase rate.

Most modern commodity switches use a dynamic α algorithm to determine the value of PFC threshold: Let α be a parameter with the range from 0 to 1, m be the total switch buffer size and m' be the amount of buffer currently occupied. For a given α, the value of t_{PFC} is dynamically computed according to the following equation $t_{PFC} = \alpha(m - m')$. During runtime, once the queue length of an ingress queue exceeds the instant t_{PFC}, a PAUSE frame will be sent to its upstream device. Note that a PAUSE frame will take some time to arrive an upstream device and take effect. To avoid packet loss due to this delay, some buffer headroom must be reserved for each ingress queue, and hence the value of m in the equation is usually slightly smaller than the total switch buffer size.

In the next, we measure the time to create a deadlock when setting different α and TTL values.

Measurement of the time to create a deadlock: We manually configure a loop between two Arista 7050QX32 switches which have 32 full duplex 40Gbps ports and 12MB shared buffer. In Fig. 3, we set α and TTL to different values and measure the time to create a deadlock under different injection rates.

We can make four observations from the results in Fig. 3: 1) It takes only a few milliseconds to create a deadlock even when the injection rate is less than 2Gbps. This indicates that it is easy for a deadlock to occur even when only a transilient loop exists in the network. In addition, we cannot rely on any loop detection and recovery solutions to prevent the occurance of deadlocks as they are too slow to resolve the loop within a few milliseconds. 2) As the injection rate increases, the time to create a deadlock decreases accordingly. 3) Given a fixed injection rate, a smaller α value requires less time to create a deadlock. 4) Given a fixed injection rate, a smaller TTL value requires more time to create a deadlock. This is because a smaller TTL value will make the packets get dropped faster in the loop, and thus more packets are needed to be injected into the loop to trigger switch to pause each other.

We repeated this experiment using many other combinations of TTL and α values and different number of switches. We found that all the results comply with what have been shown in Fig. 3.

The takeaway of this experiment is that: once there is a loop in the network, deadlock is easy to occur and very hard to prevent. In addition to a fast loop detection mechanism, we need an effective solution to detect and resolve deadlocks caused by all kinds of loops.

4 Solution

to be added.

References

[1] BGP for large-scale data centers. https://datatracker.ietf.org/doc/draft-ietf-rtgwg-bgp-routing-large-dc/?include_text=1.

[2] H. Zeng, S. Zhang, F. Ye, V. Jeyakumar, M. Ju, J. Liu, N. McKeown, and A. Vahdat, "Libra: Divide and Conquer to Verify Forwarding Tables in Huge Networks," ser. NSDI'14.

[3] 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.

[4] Y. Zhu, N. Kang, J. Cao, A. Greenberg, G. Lu, R. Mahajan, D. Maltz, L. Yuan, M. Zhang, B. Y. Zhao, and H. Zheng, "Packet-Level Telemetry in Large Datacenter Networks," ser. SIGCOMM '15.