BROCADE
A Broadcom Company

Target Credit Stall—An Investigation

Technical Brief

Table of Contents

Chapter 1: Abstract

This technical brief explores a specific cause of congestion in a Fibre Channel storage area network (SAN) that occurs when a target exhibits credit-stalled behavior (that is, “Target Credit Stall”). It provides a summary of the observed behaviors and an evaluation of potential architectural and design elements that contribute to the Target Credit Stall phenomenon.

It is common for implementations to issue link resets, disable ports, or apply other “big hammer” remedies that are also impactful to fabrics. This technical brief delves into alternative solutions with the objective of addressing this issue in a more “fabric-friendly” manner.

The Fibre Channel standards committee (T11) recognizes the problems and issues that result from Target Credit Stall behavior and has developed provisions in the standards to address this issue. The recommended techniques outlined in this technical brief discuss the developments within the Fibre Channel standards community to address Target Credit Stall.

Chapter 2: Introduction

Before we examine the characteristics of target devices and explore how these characteristics produce the Target Credit Stall phenomenon, we review the three causes of congestion in a Fibre Channel SAN.

Congestion is caused by lost buffer credits, credit-stalled devices, and oversubscription. Lost credits are caused by links with physical-layer errors. The effects of lost credits increase in severity as more credits are depleted and devices are unable to process requested data at line rate. A credit-stalled device is a misbehaving device that stops returning R_RDY signals (buffer credits) to the switch.[1] This causes the switch to stop sending frames to the device, which causes frames to back up in the fabric. Oversubscription occurs when a fabric device or link asks to handle more data than it can efficiently process, causing the surplus data to back up into the fabric.

Credit-stalled behavior is particularly concerning since the effects impact the fabric immediately. At 32GFC, a device that stops returning credit for just a millisecond stalls the transfer of 1600 frames, which is roughly 3.2 MB of data or the equivalent of three good-quality digital pictures. Devices that stall for a second inhibit the transfer of 1,600,000 frames or 32 GB of data—roughly the equivalent of eight HD movies. Furthermore, when credit stalls occur, frames in the fabric may be dropped, which leads to application-level timeouts of 30 to 60 seconds! Another way to grasp the gravity of credit-stalled behavior is to compare transferred frames to cars traversing a bridge—say the Bay Bridge in San Francisco. A 1-millisecond stall is equivalent to being stopped on the Bay Bridge for almost 9 minutes; a 1-second stall is equivalent to the Bay Bridge being out of service for about 6 days; and a 1-minute timeout is equivalent to the Bay Bridge being out of service for over a year![2]

Although Fibre Channel architecture provides rules for addressing credit return, a credit-stalled device that stops returning credits to the switch for 200 milliseconds or more causes frame drops and leads to link resets. Frame drops and link resets impact flows unrelated to the credit-stalled device, and when the credit-stalled device is a target device, it can affect a multitude of flows, degrading the performance of several applications and severely impacting the overall performance of the SAN.

Given the impact of credit-stalled behavior on fabrics and the significance of target devices in storage area networks, our goal is to provide a reasonable evaluation of the Target Credit Stall phenomenon and possible remedies for mitigating Target Credit Stall behavior.

1. For detailed information on R_RDY behavior, see the ANSI standard *Information Technology - Fibre Channel - Framing and Signaling - 4 (FC-FS-4)*.
2. Based on an average of 260,000 cars passing over the Bay Bridge in one day (see Wikipedia: https://en.m.wikipedia.org/wiki/San_Francisco-Oakland_Bay_Bridge).

Chapter 3: Observations and Recommendations

Let us examine our observations relating to the target characteristics that lead to credit-stall behavior, followed by our recommendations for mitigating Target Credit Stall with existing tools.

3.1 Observations

Credit-stalled devices cause the fabric to hold frames for excessive periods of time, which results in application performance degradation or, in extreme cases, I/O failure. The particular behavior is that the device stops returning R_RDYs (buffer credits) to the transmitting switch for tens or hundreds of milliseconds or even seconds.

Our investigation has determined that 4% of all ports show evidence of severe credit-stall congestion behavior, and of these ports, 53% are initiator devices and 47% are target devices. When severe credit stall occurs, it can cause devastating effects on the servers in the fabric. It can cause hundreds or thousands of applications to stall for nearly a minute.

Through our work with the Fibre Channel community, Brocade, a Broadcom company, has become aware of a known architectural issue that leads to the Target Credit Stall phenomenon. Under certain conditions, the resources of a storage array controller can become consumed and prevent the adapter's ability to move frames into the storage array for processing. When this happens, the adapter must hold the received frames in its frame buffers, and the adapter cannot return credit to the fabric (that is, it is unable to receive frames because all of its buffers are full). This is the condition that produces the Target Credit Stall behavior.

3.2 Mitigation Recommendations

Broadcom recommends the following steps to mitigate Target Credit Stall occurrences in your SAN:

1. Enable the Credit Recovery feature in the fabric and on end devices to eliminate credit stall due to lost credits.
2. Enable the MAPS feature with Fabric Performance Impact (FPI) monitoring to identify offending devices.
3. Enable the Slow Drain Device Quarantine (SDDQ) feature to isolate the Target Credit Stall device.
4. Consider using the Port Fencing or Port Toggling features to minimize the impact of occurrences.

Supporting these features, Brocade switching products have multiple ASIC-level counters that identify when a device is not returning credit. These counters measure the time a frame has been held for transmit because the device has zero credit, they count the number of frames dropped at an egress port because a frame cannot be transmitted due to the lack of credit, and they track the time a frame spends in the transmit queue before being sent to the end device. The Fabric OS® (FOS) software monitors these counters and generates alerts when credit delays by the end device are detected. These capabilities allow Brocade systems to identify credit-stall behavior exhibited by any device in the fabric, and, using these facilities, customers can quickly identify the offending device.

3.3 Design Recommendations

Broadcom recommends the following implementation modifications to address Target Credit Stall behaviors:

1. Make the unsolicited frame queue bigger.
2. Process the unsolicited frame queue faster.
3. Use the Frame Discard TOV to time out waiting frames.

4. Use the available FC-4 layer protocol pacing commands.
5. Deploy Fabric Notifications and utilize the “Resource Contention” Event Modifier function.

These recommendations are further described in Chapter 5 and reflect Broadcom’s commitment to constantly improving the user experience with Fibre Channel deployments. Toward that end, we are working with the Fibre Channel ecosystem to address the Target Credit Stall issue, which includes changes to the Fibre Channel standards to provide the tools to mitigate or resolve the causes of credit-stall behavior. These changes include the definition of the Frame Discard Timeout value and the addition of the “Resource Contention” Event Modifier for Fabric Notifications events.

The remainder of this technical brief examines the results of our conversations with our Fibre Channel ecosystem partners, an examination of the new tools for mitigating and resolving Target Credit Stall, and a discussion of the pros and cons for each solution mechanism.

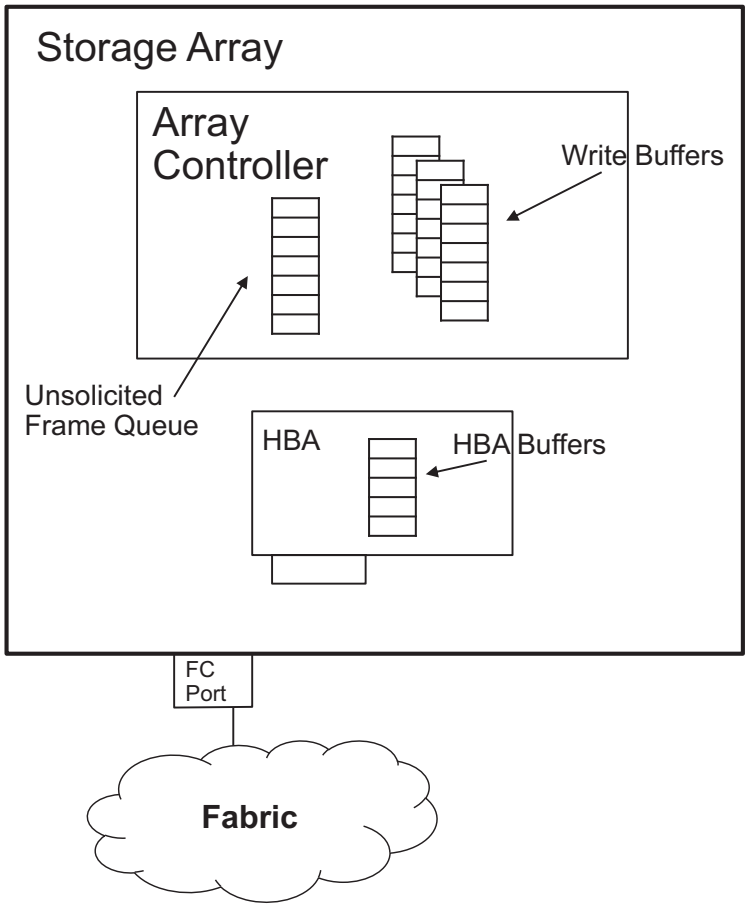
Chapter 4: Detailed Description

Let us start with a description of the basic design of a Fibre Channel target storage array device, which will help us understand the nature of the Target Credit Stall phenomenon. In particular, we have observed a specific cause of Target Credit Stall that is due to an overrun of the unsolicited frame queue. Although there may be other causes for Target Credit Stall behavior, we examine this cause and explore the effects of read/write operations on the target storage array resources.

4.1 Basic Array Controller Operation

Fibre Channel devices typically employ two primary components for SAN attachment: a Fibre Channel network adapter and an array controller (see Figure 1). As you can see, target storage arrays are essentially server-like architectures with components such as a CPU, memory, and a Fibre Channel adapter. These components interact to perform the functions necessary to implement the storage array operational characteristics.

Figure 1: Storage Array Controller



The adapter hardware (commonly referred to as the host bus adapter or HBA) provides attachment to the SAN and handles the network interface functions. The adapter uses memory to receive the Fibre Channel frames, and the amount of memory determines the number of frames that the adapter can hold. This memory is referred to as frame buffers, and the number of buffers is described as frame credits (that is, the number of frame buffers correlate to the number of buffer-to-buffer credits or BB credits).

The storage array controller provides the application “personality” of the Fibre Channel target. The array controller has data memory that is used for storing read/write buffers and has command memory that implements an “unsolicited frame queue” used for processing protocol commands into specific read/write actions. The unsolicited frame queue is generally managed as a FIFO (see Figure 1) and is the key to offloading the SAN traffic to the target.

The data and command memory are often implemented as shared memory between the adapter hardware and the storage array controller. Data is moved between the write buffers and the HBA buffers using hardware DMA functions, and the buffers are allocated based on received commands. Consequently, manipulation of the write buffers is always prearranged, and, therefore, it is unlikely that write operations will contribute to Target Credit Stall issues.[3] Commands are unsolicited events for the target device and, therefore, are managed as they arrive. The adapter hardware moves the command frames from the receive buffers to the unsolicited frame queue, and the array controller processes these commands to set up the next set of operations to be performed by the target device. If the unsolicited frame queue is not processed and the queue becomes full, the adapter cannot move the commands from the receive buffers and cannot return credit—leading to the Target Credit Stall condition.

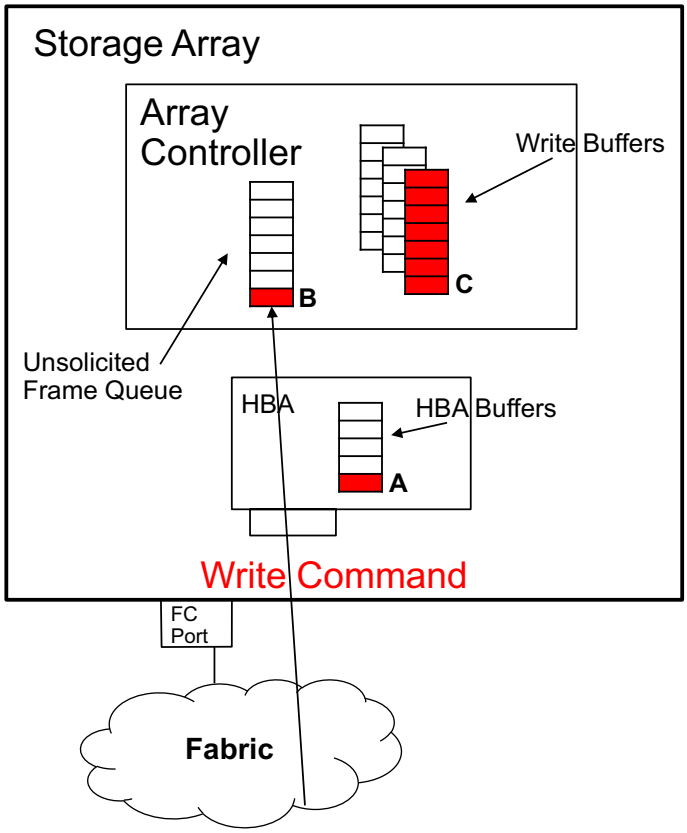
4.2 Write Operations

Next, we examine write operations as they flow through the storage array.

The operation begins with a write command sent from the initiator to the target device. The command frame is first received in the adapter frame buffers (see point A in Figure 2) and is then transferred to the array controller unsolicited frame queue (see point B in Figure 2). The command is then removed from the queue and processed, which results in the array controller allocating the write buffers to receive the write data. The array controller provides the location of these write buffers in a construct known as a “scatter-gather list” (SGL). These lists allow the write buffers to consist of disjointed memory, which provides flexibility in placing the write data.

3. Oversubscription is another cause of congestion that can involve the write buffers if the target requests more data than it can consume at line speed (see Appendix A).

Figure 2: Write Command Processing



Once the write data buffers have been allocated and the SGLs have been provided to the adapter, the array controller sends a “transfer ready” operation to commence the transfer of the write data from the initiator.[4] When the write data arrives at the adapter, the data is quickly and efficiently moved by a DMA controller to the appropriate locations identified by the SGLs (see point C in Figure 2). As soon as the contents of a frame buffer are moved from the adapter to the controller, the adapter sends an R_RDY to the switch to indicate that the buffer is available again and the switch can send the next frame.

Write operations are not typically the cause of Target Credit Stall because the target device controls the pace of the write data and effectively manages the memory used to receive the data. The target always has write buffers since they are preallocated; hence, sending write data typically does not cause Target Credit Stall as it is unlikely that the target will be overrun by write data since the data always has a place to go when it arrives.[3]

4. The term “transfer ready” references the FCP_XFER_RDY IU defined in the ANSI standard *Information Technology - Small Computer System Interface (SCSI) - Part 224: Fibre Channel Protocol for SCSI, Fourth Version (FCP-4)*.

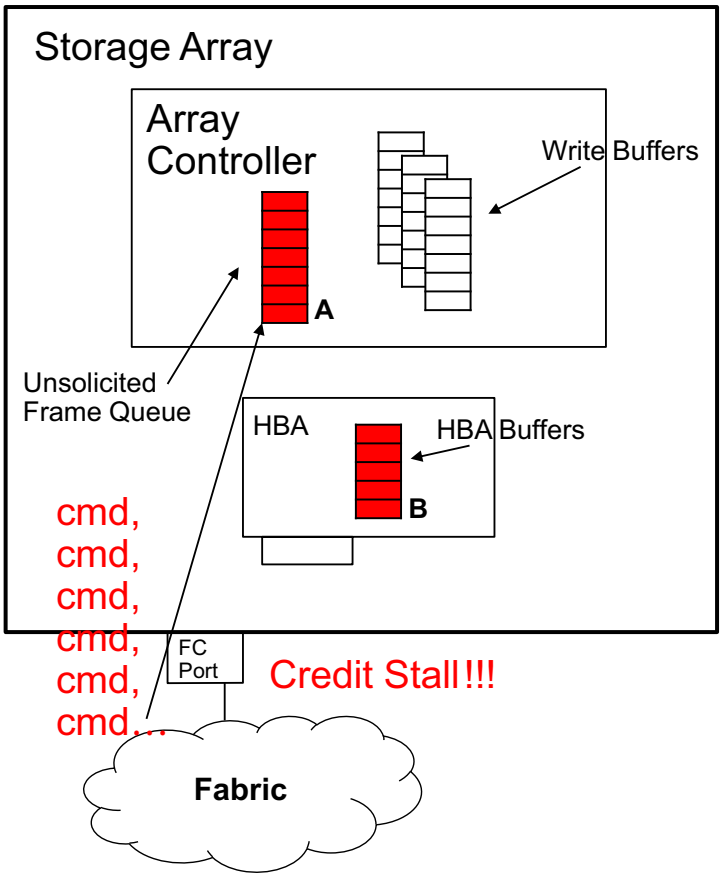
4.3 Command Processing

Now we investigate the impact of command processing as commands arrive at the storage array.

Commands are, by definition, unsolicited events. Frames that contain commands are not requested using the “transfer ready” operation as is done with write data. Instead, commands are sent by initiators without any coordination with the storage array. Therefore, storage arrays must have a mechanism for consuming, queuing, and processing commands. The most common technique is to provide a fixed number of memory buffers organized in FIFO fashion to receive the command frames, which are then used to hold the frames until the commands can be processed.

During normal operations, command frames flow efficiently from the switch to the adapter and into the controller for processing. However, if the target device resources are consumed or the command processing capacity is exceeded, unprocessed command frames can build up in the unsolicited frame queue (see point A in Figure 3). When this occurs, the adapter has no place to offload the incoming frames from the switch, and the frame buffers in the adapter fill up (see point B in Figure 3). At this point, there is no place to receive new incoming data, and the R_RDY message cannot be sent. Consequently, the switch is unable to forward frames to the device, and the frames queue up in the SAN, producing the effects of Target Credit Stall as discussed earlier.

Figure 3: Target Credit Stall Condition



Chapter 5: Architectural Investigation

We now look at some of the ideas that have been examined to mitigate the problem of Target Credit Stall due to an overrun of the unsolicited frame queue.

As described earlier, one cause of Target Credit Stall is when unsolicited frames/commands arrive at the target device faster than they can be processed. These frames are then queued, which consumes the target resources. Once the target resources are consumed, the process backs up into the adapter resources, and once the adapter resources are consumed, the process backs up into the fabric, producing the Target Credit Stall phenomenon.

There are many methods for managing this condition; we examine a few of them next.

5.1 Make the Queue Bigger

The most obvious solution to prevent the unsolicited frame queue from backing up into the adapter resources is to make the queue very large.

Modern storage arrays have lots of memory, which is generally used for caching or data management. Since the goal of alleviating Target Credit Stall is to prevent the unsolicited frames from consuming the queue, one option is to simply make the queue bigger. That is, the implementation could take advantage of the available caching memory and increase the size of the unsolicited frame queue to the point that it is unlikely to back up into the adapter resources. The queue size would be a function of the initiator-target-LUN (ITL) capacity, and the dimensions would equal the maximum number of supported initiators multiplied by the maximum number of supported outstanding requests multiplied by the maximum number of supported LUNs. Doing so ensures that there is always enough queue space to offload the frames/commands being received by the adapter.

The advantage of this approach is the simplicity of the solution (that is, simply adding more memory). The disadvantage, however, is that it can require a large amount of memory based on the intended ITL capacity of the storage array. In some cases, the quantity of memory required to support the maximums can become unrealistically large.

5.2 Process the Queue Faster

An alternative method of reducing the pressure on the unsolicited frame queue is to process the queue more quickly.

Effectively, the system prioritizes moving the queue entries from the main queue to secondary queues to be processed later at the system's leisure. This method still takes advantage of the abundance of memory typically available in modern target solutions, but it manages that memory with higher priority.

Typically, elements are not removed from the queue until an exchange context or other resource is available, but waiting for such resources is often the reason the queue backs up into the adapter. Thus, this approach removes this particular resource constraint from the unsolicited frame queue and prevents the queue from backing up in the adapter buffer memory.

Furthermore, this method also provides a technique for evaluating requests in a more logical manner relative to the load on the storage array. That is, some commands could be discarded if they are associated with a low-priority function or SLA. Those operations would then time out and be retried by the initiator. Since the retry action in this case is not associated with data delivery, it has less impact on the application.

The advantage of this approach is that it requires less memory to be dedicated to the unsolicited frame queue and it associates resource processing restrictions with more abundant, but less critical memory (for example, the offload queue). The disadvantage is that processing of the unsolicited frame queue becomes a high-priority task that may interfere with more critical operations of the storage array. Also, the Target Credit Stall condition can still be created if the offload queue becomes consumed and backs up into the unsolicited frame queue.

5.3 Time Out Waiting Frames

A variation on processing the queue faster is to time out frames that are waiting to be added to the unsolicited frame queue. This technique is further supported by the addition of the Frame Discard Timeout value (F_D_TOV) in the Fibre Channel standards.[5]

In the event the unsolicited frame queue becomes full and the adapter is unable to move commands to the queue, the frames can be discarded based on the adapter's Frame Discard Timeout setting. Effectively, the adapter limits the amount of time it waits to add a frame to the queue. If the F_D_TOV expires, the adapter discards the frame, which frees up a buffer and allows the adapter to return buffer credit to the fabric. This approach reduces the backpressure on the fabric, reduces the probability that the fabric discards frames unrelated to the storage array I/O (that is, victim flows), and allows frames to continue moving toward the storage array (albeit at a slower pace).

The advantage of this approach is that unsolicited command frames are drained off in favor of data frames; normal I/O recovery procedures are used to retransmit the discarded commands; and the queue-full condition does not prevent frame flow resulting in the Target Credit Stall behavior. In addition, the discards occur closer to the source of the problem, which aids in diagnosing the issue. The disadvantage of this approach is that it is an incomplete solution since credit stall can still occur. However, it reduces the backpressure that could lead to frame drops on ISLs, which impacts victim flows. Thus, this approach should be implemented in conjunction with other solutions.

5.4 Use SCSI Pacing Commands

The SCSI protocol defines commands that indicate when the target device resources are consumed, and it provides a technique for causing the initiators to hold off additional requests. Responses indicating TASK SET FULL or BUSY are returned by the target to the initiator to cause the initiator to reduce its I/O demand on the target. However, in early implementations, the SCSI driver in the operating system severely limited the SCSI I/O performance upon receiving the TASK SET FULL or BUSY response by reducing the SCSI queue as much as 50% for every occurrence. Furthermore, the SCSI driver did not restore the request rate until the initiator was reinitialized (often via a server reboot). Since the actions associated with these commands were considered rather drastic, most target solutions did not use them to control I/O demand and resorted to backpressure instead.

Modern initiator solutions have addressed the severity of the back-off operations instigated when receiving the SCSI pacing commands. Today, most solutions immediately reduce the I/O demand, but then gradually restore it to previous levels if no further pacing commands are received. The result is that the SCSI pacing commands produce the desired effect while adhering to the original intent of the operations. Thus, storage solutions could use the pacing operations to provide a protocol-level solution to the Target Credit Stall problem.

5. The Fibre Channel standards define the F_D_TOV as larger than the R_T_TOV and less than the E_D_TOV. Furthermore, they recommend that the value used by the end device be smaller than the fabric's F_D_TOV. As of this writing, the Fibre Channel standards committee is considering techniques that allow the fabric and end devices to exchange their F_D_TOV values as part of the Exchange Diagnostic Capabilities ELS.

The advantage of this approach is that it exploits existing constructs in the SCSI protocol that are already implemented by initiators and targets. The disadvantage is that it requires extensive testing to determine the exact behaviors of implementations in production since some customers will likely have systems that still do not respond favorably to the SCSI pacing commands.

5.5 Deploy Fabric Notifications

Fabric Notifications are a newly introduced Fibre Channel standard method of alerting devices in a SAN to the presence of congestion conditions, link integrity issues, and delivery failures. The purpose of this functionality is to provide more information regarding issues associated with I/O problems to allow the devices to recover more effectively.

The Fabric Notifications architecture allows a device to indicate when it is encountering internal resource constraints that could lead to credit-stall behavior. Specifically, the architecture defines a “Resource Contention” Event Modifier to augment the Event Type reported in the Fabric Notifications event. The purpose of this Event Modifier is to provide the reporting device with a mechanism for indicating when internal resources are consumed beyond a threshold. When this occurs, the target device can send a Peer Congestion notification event to the fabric controller with the “Resource Contention” Event Modifier set. This allows the target to tell its peers to reduce or suspend their demand on the device, which mitigates or resolves the Target Credit Stall condition. Therefore, when a storage array determines that the resources associated with processing incoming commands are being consumed, it can alleviate the condition using this form of a Fabric Notifications event.

The advantage of Fabric Notifications is that the target can notify the initiators of an impending condition before it becomes a problem in the fabric. The disadvantage is that this functionality is new for Fibre Channel and will take time to be absorbed and provided by ecosystem vendors.

Appendix A: Congestion Overview

The congestion overview presented in this appendix is from the *Brocade Fabric Congestion Troubleshooting Guide*. If you would like more information about this troubleshooting guide, please contact your Broadcom representative.

A.1 What Is Congestion?

Congestion occurs when frames enter the fabric faster than they exit the fabric. As a result, frames build up in the fabric's switches while waiting for transmission. This causes traffic moving through the fabric to slow down or become "congested." Congestion can occur on device links and inter-switch links (ISLs). Backpressure from a congested port in the fabric can cause traffic to slow down on upstream ISLs. This "congestion spreading" can cause traffic from unrelated flows that use the same ISL to slow down (that is, impact "victim" flows).

The performance capacity in the fabric is impacted because links carry data slower than they should. For example, moderate congestion could cause a 16Gb/s link to function at 4Gb/s. Severe congestion could cause that link to function at a few Kb/s or over a million times slower than its capacity. Quality of service impacts due to mild-to-moderate congestion can include transmission delays, resulting in performance degradation due to increased I/O latency. Severe congestion may result in frame loss and eventually link resets. When fabric congestion reaches a level that causes frame loss, the storage protocol I/O timeouts, such as those for SCSI or NVMe over Fabrics (NVMeOF), can take up to 60 seconds. This congestion results in severe performance degradation or application failure.

Congestion becomes apparent when an application is underperforming or, in severe cases, when it fails.

A.2 Symptoms of Congestion

Symptoms to note in a mildly, moderately, or severely congested fabric are the following:

- **Mild congestion**—The traffic load is approaching the effective bandwidth of the link or device, but credit and queue latency remains low. There is typically no impact to the application performance that reaches a level that the application or user reports, but Brocade Fabric Performance Impact (FPI) alerts may be triggered. However, these alerts could indicate that the application traffic load is increasing and may require a fabric or HBA upgrade to increase bandwidth. Action should be taken, based on the FPI alerts, before the traffic load or errors impact application performance to a reportable level.
- **Moderate congestion**—Your application is sluggish. While "sluggish" is subjective to the user, this may mean a noticeable lag in response time ranging from momentary to seconds. Congestion in the fabric is impacting the application traffic flows. This indicates that a corrective action or an upgrade to the fabric or HBA is needed to increase bandwidth capacity. As congestion increases from moderate to severe, users may notice decreases in application performance.
- **Severe congestion**—Your application has stopped functioning at an acceptable level.

A.3 Causes of Congestion

Congestion can be caused by lost buffer credits, credit-stalled devices, and oversubscription:

- **Lost credit**—Lost credits are caused by links with physical-layer errors. The effects of lost credits increase in severity as more credits are depleted. Lost credit can be initially identified by credit latency at a port or by queue latency upstream from the port. As the problem persists, frame loss due to timeouts occurs at the port or at upstream ports and can eventually lead to link resets, which occur after 2 seconds if all buffer credits are lost.

  Lost credits may affect unrelated flows in the fabric due to the congestion-spreading effect. Lost credits occur when the link between a sender and a receiver experiences errors that corrupt the receiver ready signal (R_RDY), causing the credit to be permanently lost. The R_RDY signal is sent from the receiver to the sender to indicate that it has processed a received frame and that the buffer is now available to receive a new frame. Lost credits can occur on device-to-switch links, inter-switch links (ISLs), and back-end switch ports. Lost credits prevent a transmitter from sending frames as fast as possible. This results in degraded throughput or, if all credits are lost, zero throughput on the link.

- **Credit-stalled device**—A credit-stalled device is a misbehaving device that stops returning R_RDY signals (buffer credits) to the switch. This causes the switch to stop sending frames to the device. A credit-stalled device that stops returning credits to the switch for hundreds of milliseconds or more causes frame drops and link resets. A credit-stalled device is sometimes referred to as a “slow-drain device.”

  Credit-stalled devices can be identified by credit latency or frame loss at a port. In the case of frame loss, the credit stall is long enough to cause queue latencies greater than 220 ms to 500 ms. Once frame loss occurs, application performance suffers severely and can be detected by users. If the credit-stalling behavior degrades significantly, link resets occur, indicating a credit stall for more than 2 seconds. Credit-stalled devices may affect flows unrelated to the misbehaving, credit-stalled device due to congestion spreading in the fabric.

- **Oversubscription**—Congestion due to oversubscription occurs when a fabric device or link is asked to handle more data than it can efficiently process, causing the surplus data to back up into the fabric. Congestion may also occur on ISLs when data flows from multiple devices exceed the capacity of the ISL, even though no individual device has requested more data than it can handle.

  Oversubscription is identified by queue latency on one or more upstream ports and by high-bandwidth utilization at a downstream port. Congestion from oversubscription is typically caused by a bandwidth mismatch between the source and destination ports, such as a speed mismatch when a 16Gb/s device is sending to a 4Gb/s device. This may affect flows that share the same path through the fabric.

Appendix B: Additional Target Solutions

This appendix describes other solutions that may be entertained by storage vendors to mitigate Target Credit Stall behavior.

B.1 Delay Write Handling

If an implementation is memory constrained, other methods of reducing the demand on resources can be employed.

The system can stop requesting more data by suppressing requests for write data. This can be done by holding off processing of the write requests that have been received, which postpones the acknowledgment needed to stimulate the next sequence of write frames. It can also be achieved by delaying the “transfer ready” operation, which facilitates the transfer of write frames to the array. In both cases, the solution drastically reduces the write data, which is impossible to queue in the fabric, and in turn reduces the pressure on the HBA and fabric resources to just unsolicited commands. These frames can be queued more efficiently by the HBA and fabric, which increases the capacity of the system to withstand delays in command processing typical of the Target Credit Stall condition.

This approach does not solve the problem, but it does reduce congestion and helps with flooding the problem back toward the fabric. Basically, if you think of congestion as a clogged drain, this method of “turning off the water” helps reduce the overflow caused by the clogged drain.

Revision History

Target-Credit-Stall-OT101; March 24, 2021

Added the latest additions to the Fibre Channel standards.

Target-Credit-Stall-OT100; February 19, 2020

Initial document version.