Fairness in a data center

BY

Mikkel Hagen

M.S. Computer Science, University of New Hampshire (2008)
B.S. Computer Science/Biology/Psychology, St. Bonaventure University (2004)

DISSERTATION

Submitted to the University of New Hampshire
in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

in

Computer Science

December 2012

This dissertation has been examined and approved.

Dissertation director, Radim Bartos
Associate Professor of Computer Science

Philip Hatcher
Professor of Computer Science

Robert Russell
Associate Professor of Computer Science

Chris White
Associate Professor of Mechanical Engineering

Glenn Virball
CTO, VGV Research

Robert Noseworthy
Chief Engineer, InterOperability Laboratory

Date

Dedication

To my wife Rachel and my family for all of their love and support.

Acknowledgments

I would like to first thank my advisor, Radim Bartos, for his commitment to making sure that each of my papers had the best chance of getting accepted. His steering my way through this process and his mentoring me in my burgeoning teaching career. I would like to thank my previous advisor Elizabeth Varki for all of her help in getting me to this point. She worked with me tirelessly to improve my writing and encourage me through the whole process of getting to candidacy.

I would like to thank all those at the University of New Hampshire InterOperability Laboratory for their insight, time and helpful comments/support. I would like to especially thank Robert Noseworthy and Jeff Lapak for their direct mentoring of me through this process and taking the time to review so many of my papers. I would like to thank Dan Daly from Intel, Kirt Gillum from Dell, Joel Gibson from Intel, Paul Doherty from Intel, John Fastabend from Intel, Earl Apellanes from Brocade, and Brian Shlisky from Cisco for their equipment, time, and commitment to review my papers/brainstorm.

I would like to thank my committee for their review of this document, their insights and support.

Finally, and most importantly, I would like to thank Ryan Zarick for his tireless motivation and helping me with all of my research. I would not have been able to complete this without you prodding me along and helping me to never give up.

Table of Contents

List of Tables

List of Figures

ABSTRACT

Fairness in a data center

by

Mikkel Hagen
University of New Hampshire, December, 2012

Existing data centers utilize several networking technologies in order to handle the performance requirements of different workloads. Maintaining diverse networking technologies increases complexity and is not cost effective. This results in the current trend to converge all traffic into a single networking fabric. Ethernet is both cost-effective and ubiquitous, and as such it has been chosen as the technology of choice for the converged fabric. However, traditional Ethernet does not satisfy the needs of all traffic workloads, for the most part, due to its lossy nature and, therefore, has to be enhanced to allow for full convergence. The resulting technology, Data Center Bridging (DCB), is a new set of standards defined by the IEEE to make Ethernet lossless even in the presence of congestion. As with any new networking technology, it is critical to analyze how the different protocols within DCB interact with each other as well as how each protocol interacts with existing technologies in other layers of the protocol stack.

This dissertation presents two novel schemes that address critical issues in DCB networks: fairness with respect to packet lengths and fairness with respect to flow control and bandwidth utilization. The Deficit Round Robin with Adaptive Weight Control (DRR-AWC) algorithm actively monitors the incoming streams and adjusts the scheduling weights of the outbound port. The algorithm was implemented on a real DCB switch and shown to increase fairness for traffic consisting of mixed-length packets. Targeted Priority-based Flow Control (TPFC) provides a hop-by-hop flow control mechanism that restricts the flow of aggressor streams while allowing victim streams to continue unimpeded. Two variants of the targeting mechanism within TPFC are presented and their performance evaluated through simulation.

Chapter 1

Introduction

1.1 Convergence

Data centers use different network fabrics to meet the quality of service requirements of different workload types. There are three main workload types, namely, general networking traffic, storage traffic, and inter-processor traffic. General networking traffic is handled by Ethernet, storage traffic is handled by Fibre Channel (FC), and inter-processor traffic is handled by Infiniband (IB). Each network fabric is designed to meet the throughput and latency requirements of its workload. General networking traffic is very mixed and includes web traffic along with email and socket applications, so there are few quality of service requirements. Storage traffic between storage devices and servers requires high I/O rates, high capacity and non-disruptive data delivery. Inter-processor traffic requires low latency. General network traffic does not have stringent latency requirements and is not drastically impacted by packet loss. Storage and inter-processor traffic have low latency requirements and require guaranteed delivery of packets. Figure 1-1 shows an example of a converged datacenter.

Unfortunately there is a major down-side to having several fabrics in a single data

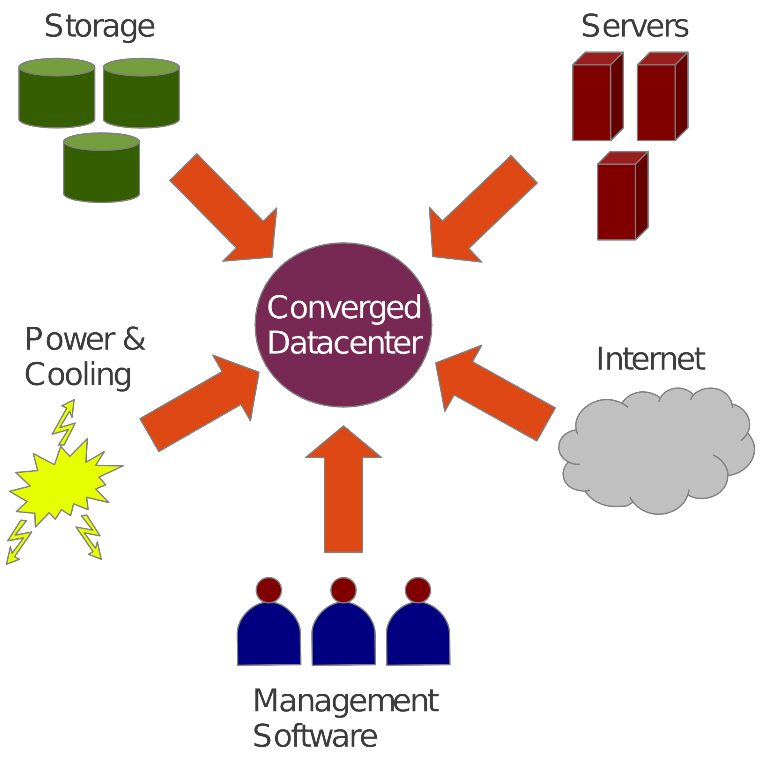


Figure 1-1: Converged Datacenter

center. There is a high cost to managing and maintaining the various fabrics in a data center. Having several fabrics in a single data center results in high heat generation due to additional heat generating hardware as well as reduced air flow, so cooling costs are high. The cost of purchasing different types of equipment for each technology is high. Furthermore, staff with various skills and expertise are needed to maintain the different fabrics, so technical and management costs rise. As a result, there is a move to using the same fabric for transmitting different workloads. The fabric must be partitioned into virtual fabrics, where each virtual fabric link is reserved for a particular workload type. Ethernet is the fabric of choice since it is the cheapest and most ubiquitous of the three fabrics. However, a major disadvantage of Ethernet is that under high traffic conditions congestion occurs and Ethernet may start dropping packets which is unacceptable for most storage and inter-processor traffic. In order to transmit different workloads on Ethernet, it is necessary to ensure that Ethernet becomes “lossless”.

Ethernet does provide flow control mechanisms to reduce packet loss during congestion. One mechanism classifies traffic workloads into eight priority levels (0-7). The lower priority traffic classes have to wait until higher priority traffic classes are transmitted, so there is less chance of packet loss in high priority workloads. Workloads with strict performance constraints, such as SAN traffic, are given a high priority. Another flow control mechanism available is Ethernet PAUSE [5]. When a computer (sender) transmits messages faster than some part of the network (receiver) can process the message, the receiver transmits a PAUSE frame back to its senders. The PAUSE message results in the sender halting transmission of data for a specified period of time. Unfortunately, the Ethernet PAUSE mechanism does not differentiate between senders, so all senders have to stop transmitting, not just the sender that was overwhelming the receiver. The Ethernet PAUSE mechanism is not sophisticated enough to handle multiple traffic classes on the fabric. With Ethernet, a specific traffic class can only avoid losing packets using higher-level protocols like TCP/IP that recover from packet loss. However, these reliable protocols have too much overhead to be useful for storage and inter-processor traffic classes. There is a clear need for a low-level, sophisticated congestion reduction mechanism that differentiates between traffic classes.

1.2 Data Center Bridging

Data Center Bridging is a new set of standards defined by the IEEE standards body to handle the smooth transmission of multiple traffic classes on Ethernet. DCB defines four protocols, namely, Enhanced Transmission Selection (ETS), Priority-based Flow Control (PFC), Congestion Notification (CN) and DCB eXchange Protocol (DCBX).

ETS provides a guaranteed bandwidth allocation for the eight traffic classes. PFC allows for the independent pausing of traffic from specific classes. For example, if a switch is getting overwhelmed by packets from a specific class, then PFC allows the switch to send PAUSE signals back to the transmitter of this traffic class. PFC is a link by link flow control control protocol, and only works between devices that are directly connected. A disadvantage of PFC's link by link behavior is "congestion spreading" — the congested node sends PAUSE frames to its sender neighbor node; the neighbor reduces its transmission rate, so the neighbor gets congested and sends PAUSE frames to its sender neighbor; in this manner, the congestion from the first congestion point spreads through the network. To prevent congestion spreading, CN allows a congestion point to send a message across the network that tells the originator of the heavy traffic to slow down. DCBX provides a mechanism to ease configuration of the network. DCBX does not affect performance on the network. It only simplifies the job of network administrators.

The DCB standards seem to have solved the problem of transmitting multiple classes of traffic streams on Ethernet. Unfortunately, the problem still persists due to a hardware shortfall, namely, switches not supporting eight traffic queues. In order to recognize eight traffic classes, switches must implement eight queues for every port. This is both cost and space expensive, so most switches only support two traffic queues. Therefore, the eight traffic classes are partitioned into two traffic queues, and each queue handles up to four traffic classes. If any traffic class starts causing congestion, the DCB protocols will throttle all the traffic classes assigned to the corresponding queue. The DCB protocols are ineffective without the supporting hardware.

Even if hardware technology improves and switches start supporting eight queues, there is another problem, namely, handling transmission of multiple streams within the same traffic class. The Internet supports several workloads - Netflix, Skype, HTTP file transfer, and BitTorrent are a small sample of typical workloads on a university subnet. Since there are far more than eight workloads, several types of workloads are assigned to the same traffic class. For example, all storage workloads could be in one traffic class. Within this class, there could be two streams. The first stream may contain packets relating to a large file's transmission, while the second stream may contain packets relating to small modifications to files. It is important to schedule the two streams so that there is minimal packet loss and some degree of fairness.

DCB is also a new technology and there is currently little research on the performance of real DCB networks. DCB drastically alters the performance characteristics of Ethernet by dividing the network into independent traffic classes that can be grouped and paused individually. It is important to understand the performance characteristics of DCB enabled Ethernet. In order to understand the benefits of DCB, it is important to evaluate the performance characteristics of each protocol within DCB and understand how the protocols interact with each other and with existing protocols. The three main protocols in DCB networks are: PFC, ETS and CN. Both PFC and ETS can be implemented with minor changes to existing Ethernet devices, so these protocols have already been deployed in devices. CN is significantly more challenging to implement and currently there are no devices that deploy it. Therefore, at this time, only simulation or analytical analysis of the CN protocol is possible.

1.3 Thesis

This thesis investigates the following questions:

- What is the throughput and latency of different applications such as iSCSI and Message Passing Interface (MPI) on a converged data center network?

  - How do the different traffic classes respond to congestion in a converged network?
  - How might the different protocols within DCB benefit a converged network?
  - What is the effect on throughput and latency when PFC is enabled?

- What is the impact of traditional scheduling algorithms, such as Deficit Round Robin (DRR), on the performance and fairness of a data center grade network?
  - What is the performance of traditional scheduling algorithms on modern data center switches?
  - How do the limitations of DRR on data center switch hardware manifest?
  - How can the limitations of DRR be resolved while:
    * maintaining the low complexity of the DRR algorithm?
    * maintaining fairness in most situations, including multiple traffic streams of different sizes and types?
  - What is the fairness of the new DRR algorithm on data center hardware as determined by well-known fairness metrics such as Jain's Fairness Index?

- How can a modified or new priority-based flow control algorithm improve fairness in a DCB network?
  - Is CN the best possible way to target aggressor streams in a network to reduce congestion?
  - Can a new aggressor stream targeting mechanism be developed that:
    * maintains the low complexity of PFC?
    * maintains the fast response time of PFC?

    * improves the ability to slow down aggressor streams while leaving victim streams unimpeded?
    * provides multiple mechanisms by which to decide what streams to target?

  - Will simulations show a marked improvement in aggressor stream targeting via the new mechanism?

1.4 Summary

This section provided an initial discussion of convergence and its benefits and limitations. To solve the limitations of Ethernet as a converged fabric, DCB was introduced. To better understand DCB, more research will need to be done in several areas including how traditional scheduling algorithms interact with modern data center hardware. Finally, a detailed thesis statement, discussing each area of research that this dissertation will be addressing, was provided.

The following chapters of this dissertation will examine each area of the thesis statement in detail. Chapter 2 provides an extensive look at existing research into the vast field of network fairness, including how it relates to DCB. Chapter 3 provides additional background on specific protocols within DCB, including challenges found when working with the protocols. Chapter 4 presents the initial look at how different traffic classes are affected by convergence and how DCB might benefit the traffic in a converged network. Chapter 5 examines initial DCB hardware and how different applications' throughput and latency are affected when PFC is enabled and disabled on the network. Chapter 6 explores the interaction of traditional scheduling algorithms, namely DRR, in a modern data center and presents a novel DRR scheduling algorithm that is shown to improve the fairness of small frames on data center hardware. Chapter 7 discusses targeting of aggressor streams in a converged network and proposes a new mechanism to target aggressor streams on a hop-by-bop basis with both low complexity of implementation and fast response time. Finally, Chapter 8 concludes this disseration, summarizing all of the findings and providing areas of possible future work.

Chapter 2

Background

2.1 Introduction

By some estimates the Internet is growing at a rate of 70-150% annually [6]. A great deal of the recent growth in internet traffic can be attributed to some of the new services, such as video/audio streaming, cloud computing and backup services. This is causing a great strain on most businesses, which need to continually expand their Data Centers, or Server Farms, to keep up with the growth. A Data Center is a facility that houses servers, switches and storage devices all in one place. Data Centers require multiple fabrics for their inter-process communication, storage and networking needs. Data Centers are the most efficient way to accomplish the computing needs of the Internet with the smallest amount of resources, including space and cost.

While being the most efficient for the task, Data Centers still have a large cost requirement. The cost can be broken up into several large areas such as cooling, direct power draw and manpower. The design of Data Centers is focused on packing maximum computing power within the smallest amount of space, which makes it difficult to keep the hardware cool. The servers are powerful and additional expansion cards are often needed for inter-process communication, storage, and networking, which all draw more direct electrical power. In order to maintain and manage a large Data Center, a team of engineers specializing in each area within the facility is needed.

Cooling, direct power draw and manpower can all be reduced by converging the fabrics within the Data Center to a single fabric. The degree of cooling can be reduced by allowing more air flow with fewer cables and by generating less heat with less hardware. The amount of direct power draw is reduced by eliminating hardware and maximizing the utilization of the existing hardware. The manpower can be reduced by eliminating different teams specialized in maintaining the different fabrics.

For several years, different technologies have attempted to converge multiple services into a single fabric. Recently the Institute of Electrical and Electronics Engineers (IEEE) has begun defining several standards which make Ethernet a "lossless" fabric. A traditional Ethernet network will often get congested and drop packets. Data Center Bridging would eliminate dropped packets due to congestion. This enables the newly defined protocol Fibre Channel over Ethernet (FCoE) to work and allows Data Centers to converge inter-process communication (iWARP and RoCE), storage (FCoE) and networking (TCP/IP) onto a single fabric.

The iWARP protocol family defines RDMA (Remote Direct Memory Access) over TCP/IP [7–13]. The RoCE protocol defines RDMA over Ethernet. In its entirety, RDMA eliminates all extra copies and allows applications to transfer data directly into the application buffer of a remote peer [14]. This has the benefit of reducing latency and maximizing bandwidth utilization. This is why RDMA has been the protocol of choice for inter-process communications, most commonly MPI (Message Passing Interface).

The FCoE protocol defines the encapsulation of Fibre Channel frames within Ethernet frames [15]. FCoE is less disruptive for existing FC installations than a wholesale removal of the infrastructure and conversion to iSCSI. FCoE has a stateless operation that allows FCoE and Ethernet on the same switch and does not have the overhead associated with TCP/IP for error control. It allows the current investment in FC equipment to be retained [16].

The TCP/IP protocol is well known and widely used. It is a connection oriented protocol that guarantees delivery of data packets. Some of the applications that utilize this protocol are email, web traffic and data backup.

IEEE 1588 is protocol defined to provide sub-microsecond synchronization on an Ethernet network [17,18]. Professional audio, industrial automation and power supply systems are some of the applications that can utilize a converged Ethernet solution through the IEEE 1588 protocol.

iWARP, FCoE, TCP/IP, and IEEE 1588 play an important role in Data Centers. Without the enhancements to Ethernet provided in Data Center Bridging there would be no control to make sure that each of the technologies would be able to work well together. Data Center Bridging consists of four different technologies. Per-Priority Flow Control, Enhanced Transmission Selection, Congestion Notification and DCB Capability Exchange. These are designed to work independently to provide enhanced Ethernet features and together they attempt to eliminate any packet loss due to congestion.

2.1.1 Priority-based Flow Control

Per-Priority PAUSE, or Priority-based Flow Control (PFC), is defined in the 802.1Qbb standard [1, 2]. It adds fields to the standard PAUSE frame that allow a device to inhibit transmission of frames on certain priorities as opposed to inhibiting all frame transmission. PFC is only defined for full duplex network interface cards (NIC) and allows link-to-link flow control on a Per-Priority basis. It is invoked by clients of the

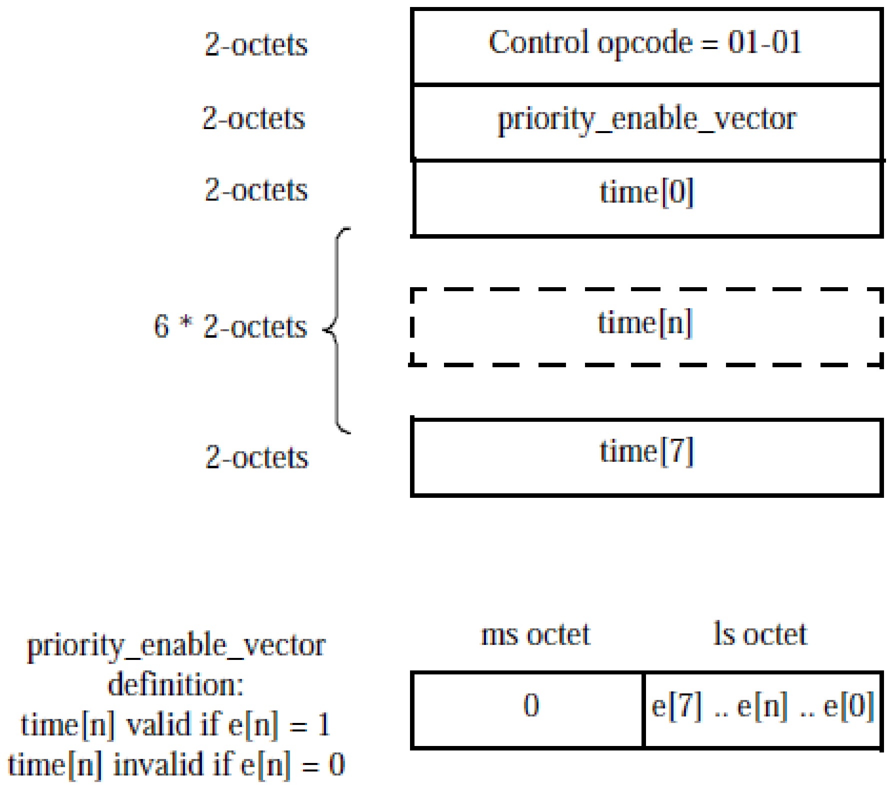


Figure 2-1: PFC frame format [1, 2]

media access control (MAC) sublayer through MAC Control PFC PAUSE primitives. PFC is used to inhibit transmission of data frames from one or more of the eight priorities found in the virtual local area network (VLAN) tag for a specified period of time. PFC cannot be used to inhibit MAC Control frames. Each PFC PAUSE frame contains an array of 8 fields containing a 2 octet priority_enable_vector field and a 2 octet time_vector field, see Figure 2-1. The priority_enable_vector field indicates for each of the eight priorities which time_vector fields are valid and should be acted upon. The time_vector fields indicate a length of time in which traffic for each priority should be inhibited. The time value is measured in units of pause_quanta, equal to 512 bit times of the particular PHY layer. A bit time is the amount of time required to transmit one bit. The pause_quanta is used so that PFC is independent of different physical layers and works the same from 10 Megabit up through 10 Gigabit Ethernet. The range of valid pause times is 0-65535 pause_quanta.

2.1.2 Enhanced Transmission Selection

Enhanced Transmission Selection (ETS) is defined in the 802.1Qaz standard [19]. ETS provides a means for network administrators to allocate link bandwidth to different priorities on a percentage of total bandwidth basis per egress port. To provide backwards compatibility with other scheduling mechanisms, ETS defines a concept called available bandwidth. Available bandwidth refers to the maximum percentage of available link bandwidth after priorities not controlled by ETS are serviced. Once allocated, a priority may only use available bandwidth up to the maximum percentage allocated.

2.1.3 Congestion Notification

Congestion Notification (CN) is defined in the 802.1Qau standard [20]. CN is a mechanism to transmit congestion information on an end-to-end basis per traffic flow, to end stations that are capable of rate limiting. A consequence of link level pausing (i.e., 802.1Qbb) is “congestion spreading” — the domino effect of buffer congestion propagating upstream causing secondary bottlenecks. A layer two congestion control algorithm allows a primary bottleneck to directly reduce the rates of those sources whose packets pass through it, thereby preventing secondary bottlenecks.

Congestion Notification is broken up into two algorithms. The first algorithm, Congestion Point (CP) Dynamics, is the mechanism in which a switch buffer samples incoming packets and generates a feedback message addressed to the source of the sampled packets with the extent of the congestion. Reaction Point (RP) Dynamics is the mechanism by which a Rate Limiter (RL) decreases its sending rate based on feedback and increases its rate without further feedback to recover lost bandwidth and probe for available bandwidth.

The CP computes a congestion measure indicating the level of congestion of its buffers. With a probability depending on the severity of congestion, the CP then selects a frame from the incoming frame buffer and sends a congestion notification message (CNM) back to the source of the frame. For example, as congestion gets higher there is a higher likelihood of randomly sampling the buffer and sending a CNM to the source of the sampled packet indicating congestion level.

RP will decrease rate proportional to the degree of congestion reported in the CNM received. Since Ethernet does not contain acknowledgments there is no feedback mechanism in which to increase rate once limited, so a timer is implemented. The rate increases in the two phases of Fast Recovery and Active Increase. In Fast Recovery (FR) the RL will increase its rate by 1/2 (Current Rate + Target Rate) every 150 kilobytes transmitted at the reduced rate if no more CNM arrive. This will occur for 5 cycles. Active Increase (AI) phase begins after 5 successful cycles of FR. During AI, the RL will probe for additional bandwidth by updating the Target Rate and

Current Rate in 50 packet cycles.

The devices in a DCB network that are configured to support Congestion Notification form what is called a Congestion Notification Domain (CND). Congestion Notification Priority (CNP) consists of one priority value such that all devices in a CND are configured to assign frames at that value to the same CP and/or RP. Different priorities coincide with different applications or even single applications. Frames with the same priority value and all assigned to a single flow queue and RP in the originating end station form a Congestion Controlled Flow (CCF). Every frame in a CCF carries a CN-tag. The CN-tag contains a FlowID. The FlowID and destination address are the means by which to identify a target of a CNM. When a CNM is created at a CP, the CP will insert the FlowID and Destination Address from the frame that is sampled from the incoming frame buffer into the CNM. As both FlowID and Destination Address are used to identify a unique flow, different end stations can use the same FlowID without a problem.

2.1.4 DCB Capability Exchange

Data Center Bridging Exchange (DCBX) protocol is defined in the 802.1Qaz standard [19]. DCBX is used to exchange information between directly connected peers in order to either detect a misconfiguration or attempt to configure each other. This allows network administrators to quickly and easily setup a new DCB network or reconfigure an existing network. DCB exchanged parameters are packaged into organizationally specific Time/Length/Value (TLV) fields and transmitted via the Link Layer Discovery Protocol (LLDP) [21]. Exchanged parameters are broken up into administered and operational parameters. Administered parameters are those that are configured by the network administrator. Operational parameters are those that are the current operational state of the device, which may or may not be the same as the administered parameters. Operational parameters can change from their original administered values due to exchanges with the peer and are only present for parameters that can be changed by the peer. DCBX is expected to operate only over a point-to-point link and if multiple peers are discovered, the peer's TLVs should be ignored until the multiple peers condition is resolved. DCBX currently only has TLVs defined for Priority Groups, PFC and Applications.

2.1.5 Summary

In order to study Data Center Bridging thoroughly, several areas need to be explored. Some of these areas include modeling of DCB networks, analysis of fairness protocols in DCB networks, and analyzing latency and bandwidth limitations in DCB networks. This chapter provides a review of the state-of-the-art in fairness and performance analysis techniques for networks, with an emphasis on Storage Area Networks, because SANs are the primary application driving DCB development. Some theoretical fairness studies are reviewed, especially a specific area of fairness referred to as Weighted Fair Queueing (WFQ) which is the most popular and flexible fairness algorithm. We will also review some of the recent performance analysis studies in FC and iSCSI.

Prior work on theoretical fairness is presented in Section 2.2. WFQ is presented in Section 2.3 because it is a popular fairness algorithm that is both simple and flexible. Section 2.5 presents an overview for the modeling and measuring techniques of the current research in storage area network performance analysis. Section 2.6 summarizes this chapter and concludes the review.

2.2 Fairness

The DCB technology allows a network administrator to break up all of the traffic flows in a network into eight traffic classes. A network administrator has the freedom to design fairness into the network. For example, the administrator can configure one traffic class to utilize most of the network or configure the network so that all traffic classes utilize an equal amount of the network.

One issue that arises from the design of the DCB technology is that the behaviour within a traffic class is undefined. While the performance characteristics between each of the eight traffic classes are well defined, it is left up to each implementation to prioritize the traffic within a traffic class. Storage area networks provide a readily identifiable, worst case example of this issue. In a storage area network, different traffic flows have drastically different performance characteristics. For example, within the traffic class priority designated for Fibre Channel over Ethernet there could be two traffic flows present. The first traffic flow could be made up of users making small changes to shared documents, which would consist of infrequent minimum sized read and write messages. The second traffic flow could be made up of a backup algorithm saving a shared file system to long term storage, which would consist of constant maximum sized read and write messages. Without a proper fairness algorithm, the second traffic flow will consume nearly all of the bandwidth provided to the traffic class.

To begin our examination of various fairness issues within a data center, a survey of the some of the latest work in general fairness is required in order to provide a baseline for further work. Most fairness algorithms can be categorized into either per-flow fairness or per-node fairness. Per-flow fairness algorithms ensure that fairness is achieved between different traffic flows within a network regardless of where they originate or are destined. This category of fairness benefits nodes with many flows.

Per-node fairness algorithms ensure that each node within a network is fairly scheduled. This category of fairness penalizes nodes with many flows on them. The field of formal specification provides us with a mechanism to define and codify fairness.

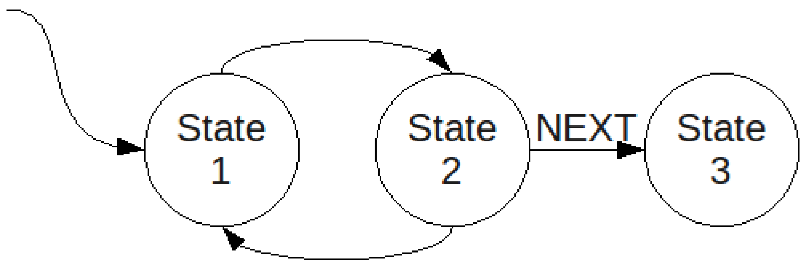


Figure 2-2: Strong and weak fairness

Strong Fairness: A scheduler is said to be strongly fair, if there are no actions that are enabled and disabled in an infinite cycle without getting a chance to act [22]. Strong fairness can be visualized in Figure 2-2. In other words, if the NEXT action is in a constant cycle of being enabled and disabled, then the action must be taken infinitely often [23]. The definition for Strong Fairness can be related to a device that attempts to transmit on a network but if the network is busy the device stops trying to transmit. Eventually the device will need to transmit its data or it will continue to accumulate more data to transmit and may overflow its buffers.

Weak Fairness: If any action remaining enabled is eventually taken, then such a scheduler is said to be weakly fair [22]. Weak fairness can be visualized in Figure 2-2. If the scheduler is in State 2 and the NEXT action is always enabled, then NEXT must be taken eventually for the scheduler to be Weakly fair [23]. The definition for Weak Fairness can be related directly to a device on a network that is attempting to send something and is prevented indefinitely from transmitting, which is also referred to as starvation.

Fairness is an often overlooked aspect of network research papers. Many studies address fairness, but do not provide a proper definition nor analysis. This may be due to the fact that packet level fairness is an NP-hard problem [24]. The definitions of Strong Fairness and Weak Fairness provide a common baseline in which to address fairness in networking. The ability to define either a Strong or Weak Fairness algorithm is highly dependent upon the network architecture and how devices transmit and receive data on the network. Therefore, an overview of some recent studies exploring different fairness mechanisms on different networks is presented.

2.2.1 Fairness in ring topologies

The most common network architecture in which fairness is studied is a ring topology. A ring topology consists of nodes connected to each other by full duplex links with the last node connected back to the first node. A ring topology has the benefit of requiring less expensive hardware than a fabric topology. The main drawback of a ring topology is shared bandwidth on the links between nodes. Several research papers examine fairness algorithms over variable ring topologies that vary the mechanism used by devices to transmit data [25–29]. Two of the more recent studies are presented below.

Resilient Packet Ring Architecture

Gjessing [25] presented a fairness algorithm for the Resilient Packet Ring (RPR) Architecture. The RPR protocol is defined in the IEEE 802.17 working group. RPR is a full duplex ring topology as shown in Figure 2-3. Nodes are called stations. A subnet connecting stations and moving traffic in one direction around the ring is called a ringlet. A full duplex RPR architecture can support up to two ringlets. Multiple packets can run on different links between stations at the same time. RPR supports only High and Low priority traffic classes. Each station in the ring has separate

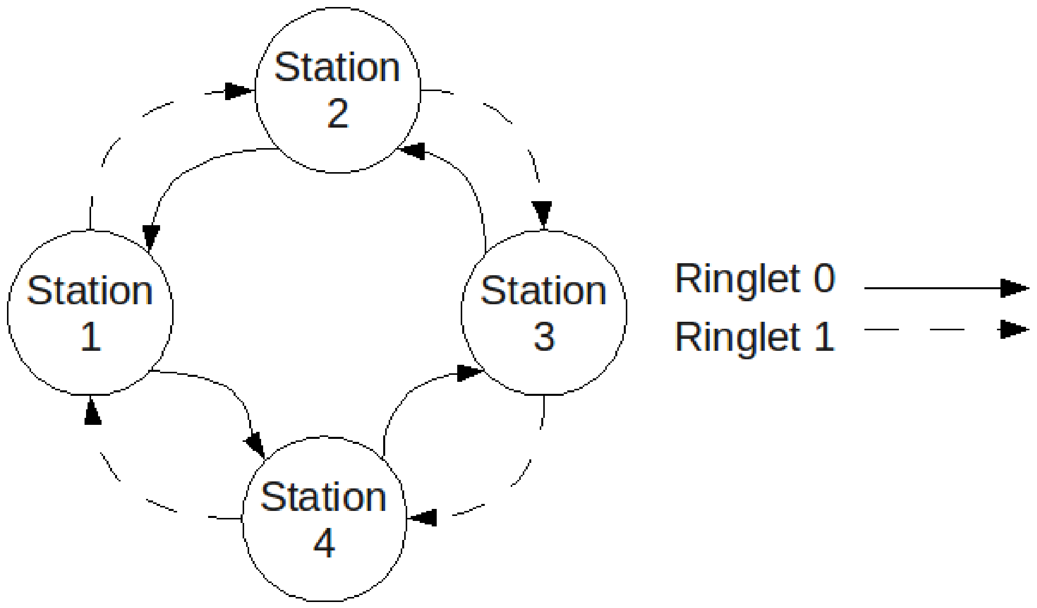


Figure 2-3: Resilient packet ring architecture

buffers for pass-through traffic and self traffic. Pass-through traffic consists of traffic that originates from other stations and is destined for other stations but needs to be passed around the ring. Self traffic consists of traffic that originates from one station and is destined for the same station.

High priority traffic is assumed to be minimal and far below the total available bandwidth, so the authors state that the fairness algorithm does not need to control high priority. The authors propose a fairness algorithm that defines packets in the pass-through buffer and self buffer to have equal priority. The station chooses a packet from each buffer every other time, or a byte count is kept if packet sizes vary greatly. Once a certain threshold is reached in the pass-through buffer, the station is not allowed to send its own packets. When a station is in this situation, it sends a congestion notification message upstream to inform the upstream sender to slow its transmission. The researchers show, using a network simulator, that this establishes a convergence of rates among stations, so that eventually they are all sending at the same rate. The algorithm proposed by the author was incorporated into the IEEE

802.17 standard as RPR Aggressive Mode fairness and became the default fairness algorithm.

Zhou et al. [29] identify an issue with RPR Aggressive Mode fairness where a permanent oscillation can occur such that a station will go from transmitting at maximum rate to not transmitting at all. They propose a fairness algorithm based on the aggregated flow of ingress traffic. Traditional fairness algorithms Max-Min and Proportional Fairness are based on flows in the network. The Max-Min Fairness algorithm is characterized by progressively maximizing the bandwidth of the flows with the lowest bandwidth requirements [30]. The Proportional Fairness algorithm is characterized by balancing the desire to maximize the total throughput of the network with the desire to allow all flows a minimum data rate [30]. The Zhou et al. algorithm breaks up the network traffic into virtual flows that are defined as continuous packets from the same source and destination. When a station becomes congested, the Zhou et al. fairness algorithm will begin forwarding the pass-through traffic and the station's own traffic fairly based on the virtual flows and the Proportional Fairness algorithm. This eliminates the oscillations and produces an overall increase in the throughput of the system.

2.2.2 Fairness in wireless networks

Another network architecture in which fairness is commonly studied is in wireless networks [24, 31]. Wireless networks are interesting in that all devices share the common medium, air, to transmit and receive data. The medium is broken up into either the time domain or frequency domain. In the time domain, devices take turns transmitting or receiving data. In the frequency domain, devices break up a large chunk of wireless spectrum into smaller parts to transmit or receive data. To further multiplex the medium, protocols can use a combination of frequency domain and time domain mechanisms to allow even more devices to transmit and receive data. A recent study exploring fairness on a wireless network is presented below.

With the deployment of commercial wireless services, fairness on wireless networks becomes an increasingly important aspect as customers will need to be guaranteed a minimum level of service. In wired networks the characteristics of the wire does not change greatly, while a wireless medium can suffer from drastic changes related to temperature, humidity or solid object interference. Nandagopal et al. [31] indicate that defining a fairness model for wired networks is fairly straight-forward, and many different fairness algorithms have been proposed for wired networks. A new type of scheme must be created for wireless networks due to features such as spatial contention for the channel, trade-off between channel utilization and fairness, inaccurate state and decentralized control.

Spatial contention refers to the use of a location's wireless spectrum by multiple devices and is in contrast to temporal contention in which multiple devices take turns utilizing the medium by breaking up the wireless spectrum based on time. The trade-off between channel utilization and fairness refers to the balance between maximizing the amount of data sent on a wireless network and ensuring that every device on the network is allowed to transmit data. Inaccurate state refers to the fact that wireless devices work without having a complete view of the entire network. Decentralized control refers to a wireless ad-hoc network where there is not a single master node that grants permission to the other nodes to transmit data.

Wireless multiple access protocols consist of collision avoidance and contention resolution. Collision avoidance is an attempt to ensure that when transmitting data no other devices are transmitting at the same time. Contention resolution has been typically achieved through backoff and persistence. Backoff consists of deferring transmission for a period of time governed by a random timer. Persistence is achieved by stations maintaining a persistence probability, or the probability that the device will transmit, and contending for the channel when it detects a clear channel. Thus, fairness is essentially an algorithm that balances persistence and backoff.

Protocols such as MACAW [32] and CB-Fair [33] use per-flow queues with per-flow backoffs. IEEE 802.11 uses per-node queue with a per-node backoff. Per-node solutions ensure that fairness is achieved between different nodes but suffer from the major problem of head of the line blocking. Head of the line blocking occurs when multiple flows are active on the same node and one flow is paused which causes all of the other flows to be blocked. This unfairly penalizes nodes that might have many flows. The authors [32] propose a fairness algorithm using a per-flow utility function. They show that system-wide fairness can be achieved without explicit global coordination as long as each node executes a contention resolution algorithm that is designed to optimize its local utility function. The authors [32] define an algorithm with three states of no_contend, contend and acquire. When a node wishes to begin to transmit it will move from no_contend to contend. Once in the contend state, it will begin to detect if the channel is clear at regular intervals. When the node detects the channel is clear, it will move from the contend state to the acquire state in which it may transmit. The authors were able to show that their algorithm was able to approximate the ideal fairness objective closely.

2.2.3 Fairness in arbitrary topologies

Fairness may be achieved by bandwidth reservation. Bandwidth reservation is not typically provided to applications, so the only way to provide minimum bandwidth guarantees is by using fairness and controlling packet loss. Historically, the first fairness algorithms utilized tokens passed around to nodes on the network. A node can transmit only after it gets a token. This is effective at guaranteeing fairness but causes massive link underutilization [34]. The next series of fairness algorithms proposed using buffer insertion and slotted ring algorithms, but these are inherently unfair [34]. Buffer insertion fairness algorithms relate to networks that break up the traffic into different priorities and set aside buffers for each priority. The data from each buffer is transmitted by priority. Slotted ring algorithms refers to fairness algorithms that relate to a ring topology in which data transmission is broken up into time slots and each device on the ring receives a fair number of slots in which to transmit data. This was resolved using the SAT control signal in the MetaRing architecture.

The SAT control signal is a frame that gets forwarded around the ring in both directions and allows a node to transmit data once it receives the SAT control signal much like the token algorithms [35]. This global fairness provides excellent equal access, but inevitably underutilizes the local networks since the network is idle for long periods while waiting for the SAT control signal. So, a stronger fairness algorithm is required which captures local regulation of conflicting nodes. Max-Min fairness solves the local regulation problem. In this algorithm a node can increase its access rate as long as it does not decrease the access rate of a node with an already equal or smaller access rate. Unfortunately, this has been proven to be non-deterministic polynomial-time (NP) hard.

In order to generalize fairness algorithms, some studies [22, 34] have attempted to define algorithms that will work over arbitrary topologies. This survey examines one of the more recent studies by Mayer, et al. [34] for arbitrary topologies. The authors [34] define fairness in terms of node access time and frequency and give a high level overview of fairness algorithms.

The algorithm proposed by Mayer, et al. assumes that convergence routing will be used. Convergence routing is defined as no packet loss due to congestion under arbitrary traffic patterns and with a single buffer per input port. It uses a global distance metric and both primary and secondary spanning tree links. A spanning tree is a tree made up of all the nodes in a network and is a common means to avoid loops while routing traffic. To define a primary and secondary spanning tree is simply selecting two different roots for each spanning tree. Packets progress through the network making sure to always make progress according to the global distance metric and can take short cuts if congestion occurs.

The researchers [34] propose an algorithm that provides fairness via local scheduling that is activated only after some predefined congestion condition is met. The algorithm utilizes two bit fixed size control messages. It is only triggered when a node cannot access the network for a given time period. The algorithm works with a back pressure approach. Fibre Channel and Gigabit Ethernet both use the back pressure approach to achieve lossless routing. Back pressure suffers from head of the line blocking and possible deadlocking. Thus, there is no possibility to provide minimum bandwidth guarantees over such networks.

When a node activates its fairness algorithm, it will send a regulate control message to all upstream nodes telling them to halt transmission after sending a quota of data. Once the activated node gets to send its quota of data it will send an unregulate control message to all upstream nodes. Because the algorithm does not maintain any error detection if a control message is lost, problems can arise. So the authors [34] propose to implement a timer in case of exceptional events such as a control message loss. Thus a node can resend the regulate control message after the timer has expired if the upstream traffic does not halt.

2.2.4 Summary

It is important to understand the fairness characteristics of different topologies because the DCB technology is not limited to a specific architecture. In defining fairness, it is important to understand the difference between Strong Fairness and Weak Fairness. Generalized definitions of both forms of fairness were presented. Several papers were surveyed that present general fairness algorithms for different topologies such as rings, wireless networks and arbitrary topologies. One of the more common ring topologies today, Resilient Packet Ring, was explored. Fairness algorithms designed for RPR focus on balancing the need to forward data from other nodes with the data that the local node must transmit. One wireless topology was examined because of its unique features. Wireless networks provide an interesting area for fairness research due to the possibility of drastic changes in the transmission medium. Finally, fairness in arbitrary topologies was presented. One common approach to providing fairness in an arbitrary topology is to supply back pressure which tells upstream nodes to slow transmission so that the rest of the network can transmit.

2.3 Weighted Fair Queueing

The Weighted Fair Queuing algorithm breaks up the data to be transmitted into weighted queues and transmits data from each queue via a fair mechanism, often a simple round-robin approach. The most basic algorithm breaks the queues into simple equal weights and transmits in a basic round-robin approach. More complex WFQ algorithms can provide non-equal weights to the queues and transmit from the queues in variations of round-robin or other means. WFQ is highly generalizable so that it can be implemented in many different networks. WFQ is also valued because of its simplicity.

An example of the basic architecture of a WFQ system can be seen in Figure 2-4.

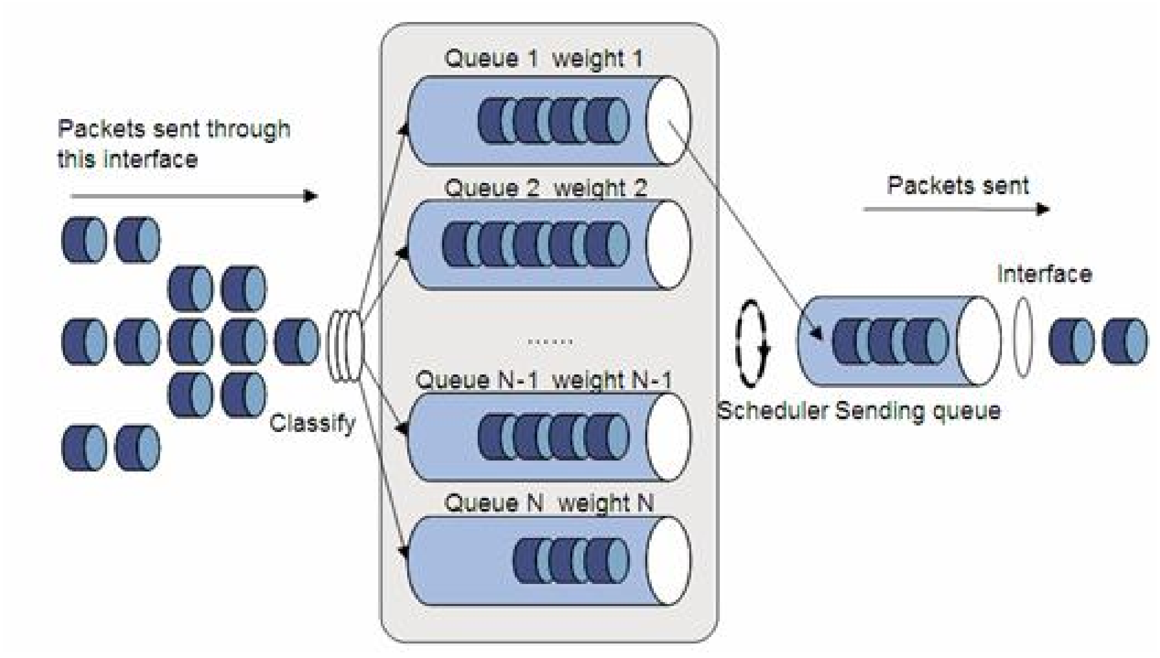


Figure 2-4: Weighted fair queuing [3]

Data packets come in from the left and initially reach the Classify unit. In the Classify unit, the packets are examined and placed in different queues. The technique that determines which queue each data packet goes into is left open and can be complex or as simple as placing different protocols into different queues. From the Classify unit, the packets are sent to their respective queues. From the figure, it can be seen that there can be any number of queues and each queue has its own weight value. The number of queues and the weight of each queue is determined by other means and largely accounts for the generalizability of WFQ. From there, the Scheduler unit pulls data packets from each queue based on the weight of the queue and a scheduling algorithm, such as simple round-robin, and places the data packet in the Sending queue. Once in the Sending queue, the data packets are sent on to their destination.

The algorithm can be altered to improve performance within specific networks. The WFQ algorithm is implemented in High Speed Wired Networks, Wireless Networks and Time Sensitive Networks. In order to develop a new scheduling algorithm for data center networks, it is important to survey the state-of-the-art in WFQ algorithms [36–45] as well as examine the more popular Deficit Round Robin which is a close approximation of WFQ.

2.3.1 High Speed Wired Networks

The challenge for fairness algorithms defined for high speed wired networks is to ensure that all of the throughput capacity is well utilized. WFQ has been shown to provide poor bandwidth utilization [40]. Several papers attempt to address the shortcomings of WFQ in regards to high speed wired networks [36, 40, 45].

Kim [40] indicates that WFQ algorithms can be categorized into the two approaches of Start-Time (ST) and Finish-Time (FT). Kim determined that FT sufficiently breaks up total traffic into different flows and provides differentiated Quality of Service (QoS) so it was used as the scheduling scheme for RSVP (Resource Reservation Protocol), which is a very popular Internet control protocol. One of the major drawbacks to the original WFQ algorithm is that it sacrifices throughput utilization for fairness. Decoupled Fair Queuing (DFQ) is one fairness algorithm that was proposed to overcome this drawback for the support of voice traffic. In DFQ, scheduling rates for flows are over-allocated and then flows are limited to the desired rate when needed.

The author [40] proposes a new algorithm, General-Time Fair Queuing (GFQ), to prevent waste by applying a general-time instead of the finish-time to WFQ. GFQ is identical to WFQ except for the use of GT instead of FT. General-time is based on a latency index to optimize latency for different flows. The authors show that GFQ was able to better utilize bandwidth by about 35% than WFQ due to scheduling more flows by means of not only the resource transformation but also resource optimization.

Yin and Xie [36] present a new WFQ algorithm called Probability based WFQ (P-WFQ). The authors identify one of the problems with the original WFQ algorithm as the calculation of the weight parameter for each packet. The proposed algorithm uses a random number to find the next packet to be serviced. It additionally groups a large number of different flows into a smaller number of groups.

Convergence of technologies into a single high speed network has created a problem of inefficient use of resources. Packet Fair Queuing (PFQ) is a large research area attempting to efficiently schedule packets at switches and approximate the idealized generalized processor sharing (GPS) policy. Some important features of GPS include delay bound, fairness and worst-case fairness. Round-robin based algorithms have the main advantage of low implementation complexity and so are predominantly used.

The proposed P-WFQ algorithm uses flow grouping, adaptive buffer manager and queue scheduling. In this paper, the researchers only consider fixed length packet systems. The algorithm generally works by grouping flows together. After grouping the flows into larger groups, each group is serviced based on a random number. Within a group, flows are serviced in a round-robin manner. The researchers were able to show that P-WFQ can guarantee a bound on the queueing delay according to a group's weight.

2.3.2 Wireless Networks

Wireless networks provide several unique challenges to fairness algorithms that are not found in wired networks. Traditional WFQ algorithms are not optimal in wireless networks due to location-dependent channel contention, spatial channel reuse and incomplete scheduling information [43]. In the worst case of highest possible contention, all wireless nodes may be connected with direct or indirect contention and any single node must consider all nodes in a global fairness model [43].

Khawam and Kofman [42] propose utilizing a wireless link by allowing the scheduler to make decisions based on a channel state. Their technique provides a dynamic reassignment of channel allocation over small time scales. Previous algorithms proposed for adapting WFQ to wireless networks have assumed that channel capacity is constant and try to make short bursts of channel errors transparent to flows. Khawam and Kofman propose Opportunistic WFQ (OWFQ) which attempts to increase the system performance through opportunistic scheduling while maintaining QoS requirements. The WFQ scheduler assigns a start tag and a finish tag to each arriving packet and serves packets in the increasing order of their finish tags. In OWFQ, scheduling is done on finish tags but is further weighted such that flows with higher quality will have a higher chance of being scheduled first. The paper used simulation to show that OWFQ provides significant system throughput gains.

Wireless ad hoc networks The Multi-hop ad hoc network is a specialized wireless network. A Multi-hop ad hoc network is a collection of nodes that communicate with each other without any established infrastructure or centralized control. At any given time a node will have frames from itself and frames from neighbors to be forwarded in its buffers. Some examples of wireless ad-hoc networks are distributed sensor networks, zero-configuration teleconferencing during disasters and data communication on the battlefield [43]. El-Khoury and El-Azouzi [37] propose a new approach to derive throughput or multi-hop routes and stability of forwarding queues. Their paper proposes to separate the frames to be transmitted from itself and frames from neighbors into two different queues, and use WFQ on both queues. These authors were able to show that end to end throughput between two nodes is not dependent on the load of the intermediate nodes. Luo et al. [43] propose a couple of variations on fair queuing for ad hoc networks. They first propose a new algorithm Maximize-Local-Minimum Fair Queueing (MLM-FQ) which identifies flows that are receiving minimal services in their immediate vicinity and ensures that they get access to the channel. They also propose Enhanced MLM-FQ (EMLM-FQ) which enhances the original algorithm by offering a larger fair share of the channel to each flow, improves scheduling resilience to collisions, and finally realizes delay and throughput decoupling which allows better utilization of the channel.

2.3.3 Time Sensitive Networks

Time sensitive applications all have a common need for the lowest possible latency. Some common time sensitive applications are Voice over IP (VoIP), streaming television, tele-diagnosing and e-commerce. During high congestion periods, problems of loss, jitter and latency become worse [41]. Several of the more recent papers relating to time sensitive fairness algorithms are explored below [38, 39, 41, 44].

Georges et al. [38] identify that Ethernet cannot be used directly for time-critical applications due to its Carrier Sense Multiple Access with Collision Detection (CSMA/CD) nature. CSMA is a simple probabilistic collision avoidance scheme in which a device will sense the shared medium and ensure that no other device is transmitting before it starts a new transmission. CSMA/CD adds a mechanism that detects if a collision occurs after the transmission begins so as to stop transmitting immediately and not waste time finishing the transmission. The collision problem can be eliminated by bringing in switches, but the problem is then shifted to a congestion problem in switches. Layer 2 Ethernet switches only provide Classification of Services (CoS) which is not true QoS. CoS only provides improved services to certain classes and no guarantee of performance. New Ethernet standards add VLAN tagging and a Priority tag but use of the standard defined Strict Priority algorithm for scheduling can result in the lowest priorities never being served. The authors [38] are able to show that the basic WFQ algorithm in conjunction with round-robin servicing is able to prevent starvation and provide an upper bound on delay that can be calculated. Further, they were able to show that lower priorities in WFQ have a lower delay and consequently higher priorities will have a higher delay.

Packet traffic on the Internet could be served on a simple First Come First Serve basis under normal traffic conditions if there was enough bandwidth. Unfortunately, it has been shown that during disasters, traffic can increase up to ten fold normal needs. During emergencies, fairness algorithms need to ensure that individual streams have reasonable latencies so several Class Based Weighted Fair Queuing (CBWQ) algorithms have been proposed. In order to better model and understand CBWFQ algorithms there needs to be accurate models and simulators. Fischer et al. and Bevilacqua-Masi et al. [41, 44] propose a new simulator for CBWFQ algorithms. CBWFQ shares bandwidth based on the weights assigned to traffic classes. It is different from WFQ in that weights are assigned to traffic classes instead of individual flows. Classes are served with a probability based on weight assigned to the class. Weights are determined based on the algorithm. For example, algorithms that favor time sensitive applications will provide them with higher weights than other traffic classes. Within a class, packets are served on a first come first serve basis. This is termed a "random polling model" as opposed to a "cyclic polling model" in which queues are served one after another in a circular fashion. The authors indicate a large amount of research has been done on cyclic polling models while random polling models have received little research attention. The authors conclude the paper by comparing their simulator to a commonly used commercial simulator. They showed that their simulator provided more efficient analysis capabilities than the commercial simulator by having shorter run times.

One important aspect of time sensitive applications is the ability to configure fairness dynamically. Wang et al. [39] first considered WFQ, but determined it was not ideal due to the inability to dynamically change the weight assigned to individual flows so as to accommodate allocation on demand. The authors propose an Adaptive WFQ algorithm (AWFQ) that essentially determines if the arrival rate of packets is greater than the amount allocated and adjusts the weight accordingly. The authors showed that AWFQ is flexible to variable bandwidth requests and prevents starvation to the best effort class of traffic.

2.4 Deficit Round Robin Scheduling

Weighted Fair Queuing (WFQ) [46] is a well known scheduling algorithm that maintains fairness while providing good performance. Unfortunately, WFQ is expensive to implement in hardware, and has a complexity of O(N), where N is the number of workloads to be scheduled. As a result, most switches implement the Deficit Round-Robin (DRR) [47] scheduling algorithm. DRR is comparable to WFQ in fairness and has a lower complexity of O(1). We explain the DRR algorithm using Figure 2-5, which presents an example system with three input queues on the left and a single output queue on the right.

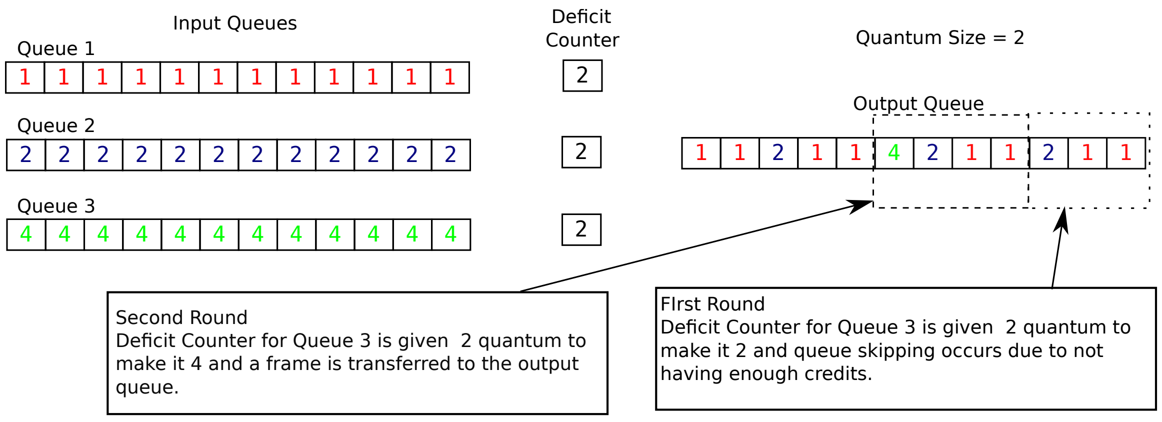

Input Queues
Queue 1
1 1 1 1 1 1 1 1 1 1 1 1
Queue 2
2 2 2 2 2 2 2 2 2 2 2 2
Queue 3
4 4 4 4 4 4 4 4 4 4 4 4
Deficit Counter
2
2
2
Quantum Size = 2
Output Queue
1 1 2 1 1 4 2 1 1 2 1 1
Second Round
Deficit Counter for Queue 3 is given 2 quantum to make it 4 and a frame is transferred to the output queue.
FIrst Round
Deficit Counter for Queue 3 is given 2 quantum to make it 2 and queue skipping occurs due to not having enough credits.

Figure 2-5: DRR example

Frames that arrive at the switch are assigned to one of the three input queues. Each input queue is FIFO and holds frames relating to specific workloads. The scheduling algorithm determines in what order frames are placed in the single output queue. The frames are transmitted in FIFO order from the output queue. The numbers in each input queue represent the frame size of the frames in the queue. The algorithm scans each input queue, starting with queue 1, queue 2, and then queue 3. During each scan, zero or more frames from each input queue are moved to the output queue. The number of frames moved from each input queue to the output queue is determined by 3 variables, namely, quantum size of the algorithm, the deficit counter value of the queue, and the size of the frames in the queue. The algorithm has a fixed quantum size; for this example, the quantum size is 2. The deficit counters are set to 0 initially. The quantum size is the number of credits that accumulate each round; for this example, it is equivalent to units of frame size. At the start of each round the quantum size is added to the deficit counter for each input queue. The deficit counter for each input queue maintains the number of credits placed into it for each round along with any unused credits from previous rounds. Once each deficit counter has been incremented with the quantum size for the current round, the scheduling algorithm begins to move frames from the input queues to the output queue in consecutive, round-robin fashion. A frame is moved from an input queue to the output queue when the deficit counter is greater than the size of the frame at the head of the queue. Once a frame is moved from an input queue to the output queue, the deficit counter is decremented by the size of the frame that was moved. In the example given, the dotted line boxes on the output queue represent each round of the scheduling algorithm. With a quantum size of two, it can be seen in the first round two one unit frames were moved into the output queue along with one two unit frame. In the second round, the deficit counter of Queue 3 remembers its leftover credits from the first round and one four unit frame is transmitted with more one unit and two unit frames. This example shows how smaller quantum sizes can cause queue skipping to happen, such as with Queue 3. Thus, it is recommended to set the quantum size no smaller than the largest frame that can be transmitted.

2.4.1 Summary

Weighted Fair Queuing is a specific algorithm that is both balanced and general enough to work on many different networks. The bulk of research in this area focuses on fine tuning aspects of the WFQ algorithm to maximize the performance of specific applications. WFQ research focusing on High Speed Wired Networks, Wireless Networks and Time Sensitive Networks was presented. High speed wired networks are characterized by high throughput, but traditional WFQ is not the most optimal for utilizing network throughput. Wireless networks are characterized by limited throughput and a constantly changing channel, so proposed algorithms focus on dynamically changing flow weights in WFQ to accommodate the channel information. Time sensitive networks are characterized by a need for low latency or deterministic latency, so the proposed algorithms focus on guaranteeing traffic flows a fixed amount of bandwidth and ensure that none of the flows starve. In addition to the previous work on WFQ over various networks, we examined DRR which is a close approximation of WFQ that most current data center switches are now implementing.

Previous work in scheduling algorithms has been based on assumptions that no longer hold for a network based on the DCB technology. The ability of a DCB based network to avoid packet loss due to congestion eliminates the most important assumption that previous scheduling algorithms make. Existing scheduling algorithms do not consider maintaining fairness within a traffic class when the entire class may only have a portion of the network bandwidth or may be paused and unable to transmit for a period of time. For this reason, a new scheduling algorithm will need to be devised for the data center networks.

2.5 Performance

Storage area networks are the predominant application that is driving the development of the DCB technology, specifically Fibre Channel over Ethernet (FCoE). FCoE is a new technology that is defined to encapsulate Fibre Channel frames within Ethernet MAC frames. FCoE does not have a reasonable mechanism defined to handle packet loss, so it requires modifications to Ethernet defined in the DCB technology to avoid packet loss due to congestion.

iSCSI running over a DCB based network is referred to as Enterprise iSCSI (eiSCSI). While iSCSI does not need the DCB technology because it handles packet loss with the TCP protocol, avoiding retransmission of packets can benefit the performance of iSCSI greatly. For this reason, iSCSI is also a major application driving the development of the DCB technology.

It is important to understand the performance characteristics of both Fibre Channel and iSCSI because they are two of the strongest drivers defining the DCB technology. Both technologies are commonly categorized by large and long transfers of data. For this reason, a survey of some recent storage area network performance analysis papers are presented.

Traditional network based storage suffers from latency issues, since SCSI disk subsystems are slower than FC and neither the network nor the SCSI bus are fault tolerant. Storage Area Networks (SAN) consist of hosts, storage devices, interfaces, hubs, and switches [48]. Some of the data currently stored on SANs are storage consolidation, data replication, backup and recovery, simulations, modeling, Internet and intranet browsing, multimedia, transaction processing, e-business, and data warehousing and mining [48,49]. Power outages and natural calamaties such as earthquakes and terrorist attacks have created a need for SANs to move further away from access sites. Fibre Channel, iSCSI, FCIP and iFCP have all been introduced to transport SCSI requests and responses over great distances [50]. This section will examine several papers that are focused on modeling SANs followed by some recent papers related to two of the more popular SAN technolgies of Fibre Channel and iSCSI.

2.5.1 SAN Modeling

Analyzing and modeling a SAN before deployment is important to ensure that the best choice for each specific implementation is made. Molero et al. [49] proposed a tool for simulating the behavior of SANs using high-speed LAN interconnects. The simulator focuses on SAN internal design and main implementation features. The simulator is designed with a wide range of input parameters, such as locations for both server and disks, network topology, switch architecture, routing algorithms and maximum packet size. The authors use real world traces with the simulator to show that latency variations affect control messages more than data messages, and links generally have low utilization.

Telikepalli et al. [50] proposed models using network and protocol specific variables. They identified the network parameters of distance, packet loss, available bandwidth, advances in TCP implementations and increased maximum TCP window, which control the amount of data in flight at any given time. They also identified the protocol parameters of data window, frame size, zero copy, TCP processing delays, I/O writes, I/O block size and command sequence. Performance analysis showed that the parameters that have a key impact on performance were packet loss and buffer space at the sender and receiver.

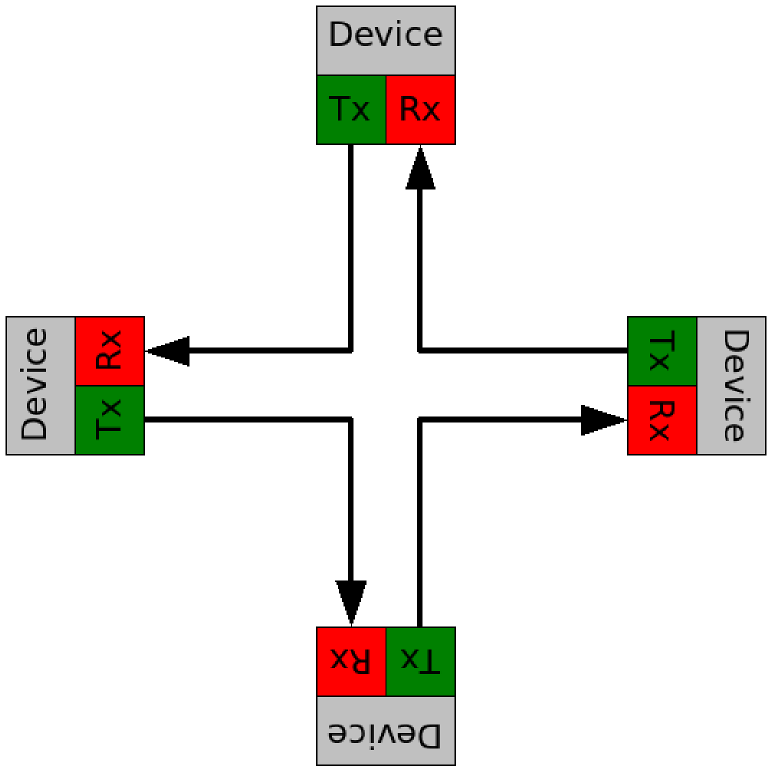


Figure 2-6: Fibre channel arbitrated loop topology [4]

2.5.2 Fibre Channel

Fibre Channel is the SAN technology with the largest market share. Fibre Channel is flexible and supports point-to-point, arbitrated loop and switched fabric topologies. The arbitrated loop and switched fabric topologies are highly scalable. The biggest drawback to deploying Fibre Channel networks continues to be poor interoperability [48]. Another key challenge is dynamically restricting access to files, which is being addressed by new zoning standards. Dynamically restricting access to files provides a baseline security measure to network administrators so that they can dictate what files each user has read and write access to. Fibre Channel loop topology was introduced as a low cost alternative to the switch design [51]. The Arbitrated Loop topology allows up to 127 active devices, including one fabric element or 126 devices if no fabric element is present. The Arbitrated Loop protocol requires that before data are transferred, the sending port must acquire loop access and establish a connection with the receiving port.

An example of an Arbitrated Loop topology is shown in Figure 2-6. Figure 2-6 shows the transmit port (TX) of each device connected to the receive port (RX) of its neighbor creating a loop. In this example, data would progress around the loop counter-clockwise.

Fibre Channel defines three classes of transmission service. Class 1 provides dedicated full-duplex circuit-switched connection. Class 2 provides a frame-switched connection with acknowledgement and flow control. Class 3 provides a connection-less datagram service with no acknowledgement. [51]

Heath and Yakutis [51] focus on examples of SAN applications such as telecommunications billing and online database transactions. They both are characterized by small data transfers of between 2 and 8 KB. The authors show that Arbitrated Loop topologies achieve throughput levels adequate to support online transaction processing (OLTP) applications. The authors also show that a dual Arbitrated Loop topology can saturate the system in the case of long I/O transfer applications.

2.5.3 iSCSI

Internet Protocol (IP) based storage networking is preferred over other SAN technologies because of economy and convenience. Two early IP storage based protocols were encapsulating SCSI over TCP/IP and SCSI and IP protocol conversion at a specialized switch [52]. Both protocols have severe performance issues due to overhead and congestion.

In a typical iSCSI network, Xubin et al. [52] found that 58% of TCP/IP packets were very small (less than 127 bytes long). The authors proposed a new cache scheme called iCache. Using non-volatile RAM, the authors created a two level hierarchical cache for iSCSI systems. iCache converts small packets into large packets, and is completely transparent to the OS. The authors found that iCache improves the iSCSI performance by an order of magnitude for 90% of storage requests, while the average speedup is about 85%.

One of the main challenges for iSCSI is its efficiency and performance against existing technologies such as Fibre Channel. Aiken et al. [53] compared the performance of iSCSI and Fibre Channel. They found that commercial iSCSI software compared quite favorably with Fibre Channel. The performance of open-source iSCSI was very dependent on the disk throughput. The researchers showed that performance was up to 50% less with disks versus no disks. iSCSI also benefits greatly from specialized hardware. The authors examined both network layer parameters and physical layer parameters. In a typical local area SAN, they found that Ethernet frame size and other physical layer parameters play a significant role in performance while network layer parameters such as TCP window buffer size have little effect on performance. Using a wide-area network emulator, the authors showed that network layer parameters played a dominant role compared to physical layer parameters in regards to performance.

2.5.4 Summary

Storage Area Networks are becoming more popular with the explosion of data intensive applications. SANs are both scalable and reliable. SANs are a large capital investment, so it is important to model the system prior to the investment and evaluate performance limitations. Performance models help in the purchase decision of what SAN technology to invest in. Fibre Channel and iSCSI are the two SAN technologies with the largest market share. Fibre Channel is a flexible and reliable SAN technology with the largest deployment. Part of the reason for its popularity is the flexibility provide by its Arbitrated Loop topology. IP based SANs are deployed due to their convenience and accessibility. iSCSI was shown to perform quite favourably when compared to Fibre Channel with both commercial products and open-source products.

FCoE and eiSCSI are two variations of the traditional Fibre Channel and iSCSI storage area networks that have become available with the introduction of the DCB technology. In order to define a new fairness algorithm that affects the performance of traffic flows within a traffic class, it is important to examine the performance differences between the different technologies. This survey showed that Fibre Channel is the more popular technology of the two and had better performance, but iSCSI performance is catching up.

2.6 Background Conclusion

There is a growing number of data intensive applications requiring more data storage. The market for data storage is large and will continue to grow. Storage Area Networks are becoming an invaluable resource for nearly every organization because they are both scalable and reliable. Moreover, Ethernet based SANs are convenient and accessible. In order to increase the value that Ethernet can provide to organizations utilizing converged networks, the IEEE has defined a series of standards in the Data Center Bridging Working Group. The working group has developed the new standards for Priority-based Flow Control, Congestion Notification, Enhanced Transmission Selection and DCBX. In their entirety, the standards make Ethernet "lossless", by ensuring that packets do not get dropped when there is congestion.

This chapter examined some of the previous work in the large field of fairness and performance. It is important to understand how the new DCB protocols affect the performance and fairness of applications such as storage and inter-processor communication over Ethernet. The following chapters will develop new understanding and mechanisms based on the foundation of previous work laid out in this chapter.

Chapter 3

Testing Challenges of Data Center Bridging Networks

After examining previous work on fairness and an introduction to the technologies that we will be examining throughout this dissertation, this chapter examines in detail each of the protocols within the DCB standard. We investigate challenges in measuring and testing devices that have implemented the protocols and provide examples for several issues that have been seen in DCB devices. It is important to understand the intricacies of testing and measuring the protocols before beginning to analyze performance and identifying bottlenecks, so that experimental results are not attributed improperly to device specific issues.

3.1 Introduction

In the Data Center, a primary driving force is the desire for network convergence. The notion of one dedicated network for storage, one dedicated network for Inter-Processor Communication, and one dedicated network for general data networking has long ago been abandoned. The many reasons include: reducing the expertise needed to maintain the data center; reducing the power consumed by (and thermal load generated by) so many disparate network switches and host channel adapters (a.k.a., network cards in the servers); reducing the space required; and reducing the overall cost of the data center equipment.

Ethernet's ubiquity drives a natural desire to see this technology expand to support the needs of the data center, but to do so, it needed to grow to support a lossless layer-2 data transport, while remaining cost effective. One of the major applications that is benefiting from convergence to Ethernet is storage. Fibre Channel is currently the predominant technology in the storage space. Storage as an application class typically utilizes a network with large, long transfer patterns. In order to converge Fibre Channel onto Ethernet, extensions to Ethernet needed to be introduced in order to handle the congestion introduced with storage transfer patterns. In large part, this need for extensions to existing Ethernet bridging drove the work in the IEEE 802.1 Data Center Bridging Task Force to develop collectively what has come to be known as Data Center Bridging (DCB).

DCB consists of four different technologies: Priority-based Flow Control, Enhanced Transmission Selection, Congestion Notification and DCB Exchange. Each technology works independently to provide enhanced Ethernet features and together they attempt to eliminate any packet loss due to congestion. DCB builds upon the existing Ethernet priority field in the VLAN tag, which provides up to eight different priorities. There are two additional technologies being defined by the Data Center Bridging Task Group within IEEE to address the growing use of server virtualization in the Data Center. These technologies are called Edge Virtual Bridging and Bridge Port Extension, but these are still in development, not widely implemented by device manufacturers and testing is also not yet developed; hence, this chapter will focus on the previously mentioned four DCB technologies. For a more thorough examination of the details of each protocol within DCB, please review the tutorial on the

University of New Hampshire InterOperability Laboratory website [54].

There are significant new Ethernet testing challenges introduced with DCB. Each of the four protocols must be tested individually, the interaction between the protocols needs to be tested and, finally, the interaction of existing applications with DCB will be important to test. When considering the process of testing a new technology it is useful to think in terms of *conformance* and *interoperability.* Conformance testing entails measuring a device's behavior as it pertains to the standard defined protocol behavior. Interoperability testing consists of connecting multiple devices within a technology space and observing the protocol's interactions with each other. It is important to consider both conformance and interoperability because full conformance may not guarantee interoperability and current interoperability does not necessarily indicate future interoperability.

While it is necessary to perform both conformance and interoperability testing, both are not always performed as part of a collaborative network testing effort. Some examples of new technologies that have taken very different approaches to testing include IPv6 and Wi-Fi. As a result of significant interoperability problems with early implementations [55], IPv6 early on developed a certification program that includes both conformance and interoperability in order to address real limitations in the technology [56]. The industry understood the importance of testing and developed both the IPv6 Ready Logo program as well as the IPv6 Testing Project [57]. Both programs combined have greatly increased the success of IPv6 in the market.

Another example of a technology with a well known testing program is Wi-Fi. Initially there were many examples of failures to successfully set up a wireless network [58]. As a result, Wi-Fi testing from the very beginning focused on interoperability over conformance. One of the issues with an interoperability only program is development and maintenance of a test bed of devices [59]. Aside from the dif-

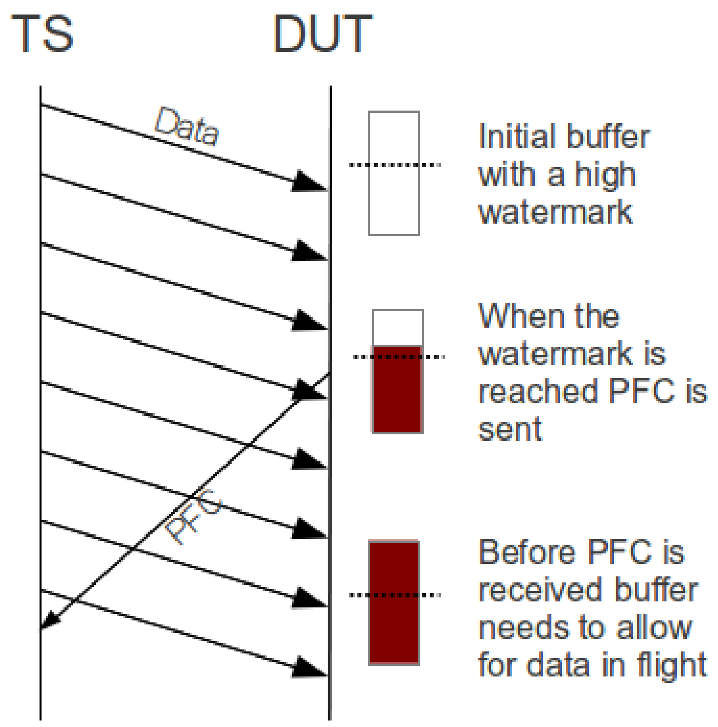

TS
DUT
Data
PFC
Initial buffer with a high watermark
When the watermark is reached PFC is sent
Before PFC is received buffer needs to allow for data in flight

Figure 3-1: Priority-based flow control space time diagram

ficulty of maintaining the testbed, another issue is the initial cost which can easily range from \$20,000 to \$40,000 [60]. Such interoperability-only approaches also tend to suffer from sensitivities to any non-conformant behavior in the 'golden devices' represented in the interoperability testbed.

DCB, IPv6, Wireless and any new networking technology all face the challenge of providing the balance of conformance and interoperability network testing while doing so in a timely and cost-effective manner. By detecting issues early, market acceptance delays can be avoided by mitigating negative experiences of end-users. This trade-off of cost vs. coverage vs. timeliness is but one of many challenges facing the adoption of DCB. The rest of this chapter will examine each protocol within DCB and highlight aspects of how to test the protocol, including early observations of common issues. Finally, this chapter will discuss applications that run over DCB and how to test them and their interactions with DCB.

Before delving into the details of testing the specific protocols within DCB, there are immediate challenges related to testing DCB across all of the different possible interfaces. In the 10G Ethernet space there are at least four common interface types which complicate the testing including: 10GBase-CX4, iPass, 10GBase-SR, 10GBase-LR and 10GBaseT. Each of these interfaces requires different test gear and adapters in order to test them as well as complicating interoperability testing if the devices in the test bed do not have the same interfaces. Additionally, DCB is being defined as speed independent with announced DCB products including: Gigabit, 10 Gigabit, 40 Gigabit and 100 Gigabit. Each of these speeds have additional interfaces and require different test stations in order to test them (all increasing the cost of test) as well as different timing conformance requirements.

3.2 Conformance

3.2.1 Priority-based Flow Control

PFC is defined in the IEEE 802.1Qbb standard. As PFC only operates on a link by link basis, it can be tested with a simple network topology. In this topology, a device under test (DUT) is connected directly to a test station. The test station stimulates the DUT with positive and negative responses and the results are observed. A test suite has been defined [61] that breaks the testing into several groups including proper behavior as well as DUT response to some common malformed frames. This testing has identified several interesting issues with early implementations of devices.

In the first test group, which examines a DUT's ability to receive properly formatted PFC frames, there is a test that examines the ability of devices to pause each of the eight priorities independently. Many early implementations failed this test because they only implemented two lossless classes. In group three which examines the DUT's proper transmission of PFC frames, there is a test that examines the worst case scenario of frames that could be in transit after a device sends a PFC frame. When a device begins to become congested it needs to send a PFC frame to its neighbor with enough free buffer space to store the frames that could already be in the process of being sent to it (bits that are "in-flight" on the wire) as seen in Figure 3-1. This ensures that the device does not lose any frames even after it sends a PFC frame and receives a limited additional amount of data. Early testing results from this test have discovered that many devices do not provide enough buffer space to ensure frame loss does not occur. The ability to monitor bit-level transmissions to/from the DUT enable enhanced accuracy of these measurements. Such testing requires an FPGA (field programmable gate array) that can insert PAUSE frames into very specific points in a traffic stream, as well as capture and monitor at the bit-level. This has both a high knowledge cost as well as high development time.

3.2.2 Enhanced Transmission Selection

Enhanced Transmission Selection (ETS) is defined in the IEEE 802.1Qaz standard. The testing procedure for ETS [61] is divided into end device and switch testing. End device testing is somewhat limited since there is only one connection to the device. Additionally, early devices from storage vendors have mapped all non-storage traffic into a single priority. This mapping only allows up to two different ETS bandwidth groups to be tested: storage and network traffic classes. Another limitation of early devices is that many switches are only supporting two different queues, so all eight priorities can only be broken up into two different bandwidth groups. Even with these early limitations ETS can provide a useful mechanism to configure network bandwidth.

ETS is tested by varying the bandwidth percentages of each of the traffic classes. Before testing, each traffic class needs to be "baselined" by itself. This involves simply

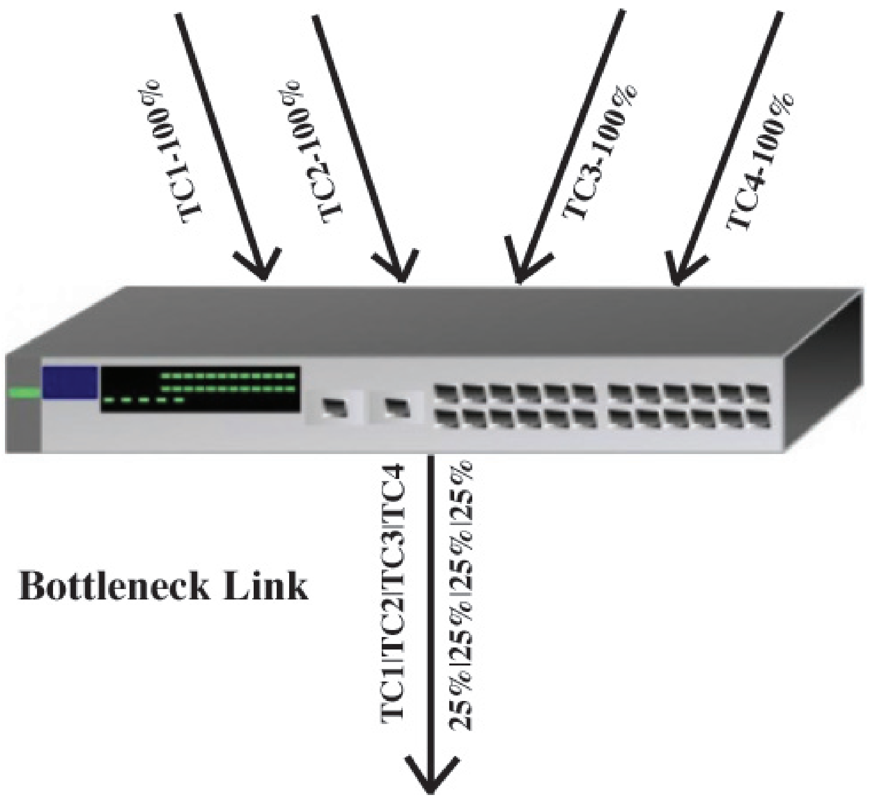


Figure 3-2: Bottleneck topology for ETS testing

running each traffic class on its own and verifying that all of the traffic classes can achieve enough throughput in order to test the different ETS percentages defined in the test plan. Since storage is one of the major applications, this is especially challenging if the storage devices cannot achieve close to line rate of 10G Ethernet.

It is important to test that ETS is work-conserving, so that if a traffic class is not utilizing its assigned bandwidth then the other traffic classes utilize the unused bandwidth. Multi-port switch testing is more interesting and can be tested using the standard bottleneck topology in Figure 3-2. Using this topology a switch's ETS algorithm can be fully stressed. The standard bottleneck topology has multiple inbound connections leading into the switch with one outbound connection. By transmitting line rate traffic into each of the input connections with a different priority for each connection, the output port is fully stressed. Utilizing all eight traffic classes supported in the DCB standard, the output port could be receiving 8 times the traffic that it can transmit and needs to schedule the traffic down to a single link while managing the assigned ETS bandwidths. The important testing criteria defined in the standard is to show that the actual bandwidth does not deviate more than ±10% of the assigned bandwidth. This could cause any device that is close to the 10% margins to become very disruptive in the network. This has caused issues in larger DCB networks where multiple marginal devices together can cause significant variation from the expected performance.

3.2.3 Congestion Notification

Congestion Notification (CN) is defined in the IEEE 802.1Qau standard. An example of the Congestion Notification mechanism can be seen in Figure 3-3. This figure shows that two flows create congestion at the third switch which could cause congestion spreading. If the Congestion Point (CP) sends a Congestion Notification Message

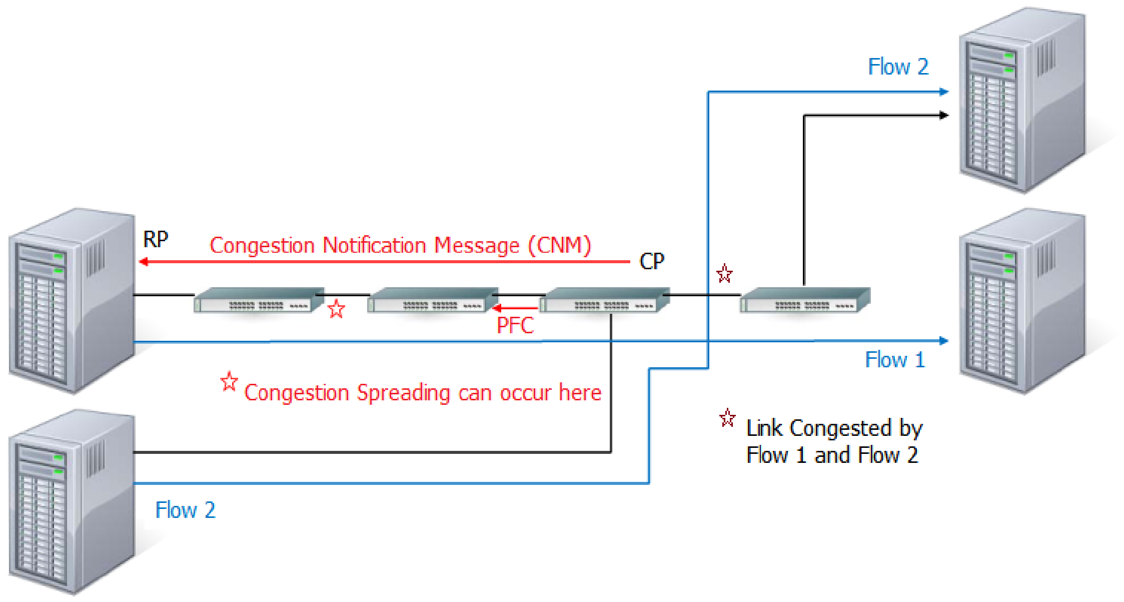

Flow 2
RP
Congestion Notification Message (CNM)
CP
PFC
Flow 1
Congestion Spreading can occur here
Link Congested by Flow 1 and Flow 2
Flow 2

Figure 3-3: Congestion notification example

(CNM) back to the Reaction Point (RP) the use of PFC can be avoided and stop the congestion spreading.

CN is the most complicated and novel of all of the DCB standards. It does not build on any existing Ethernet technologies; as a result not many vendors have implemented the CN standard yet. Despite that, the CN standard was the first standard to be completed by the IEEE working group and a test suite [61] was available early. CN testing utilizes the simple conformance topology that PFC used. The testing involves sending congestion notification messages to the DUT and observing that its frame transmission rate decreases appropriately as well as causing congestion in the DUT and observing that the DUT sends a proper congestion notification message. Since not many devices have implemented the CN standard, the test procedure has not been executed yet and test results have not yet been gathered.

3.2.4 Data Center Bridging eXchange

The Data Center Bridging Exchange (DCBX) protocol is defined in the IEEE 802.1Qaz standard. Although DCBX does not have any direct effect on performance issues, previous well attended industry test events have shown that it is the single largest piece that contributes to observed non-interoperable setups. Prior to the IEEE defining a standard for DCBX, a group of vendors contributed an early proposal that has become known as the Baseline version. Once the IEEE began to work on a standard, the new version became known as IEEE Draft version of DCBX.

The Baseline version and IEEE Draft version are completely non-interoperable. Currently many vendors are still supporting the Baseline version which is causing serious interoperability issues similar to the early days of IPv6 as seen in the VTHD network [55]. An IEEE Draft version test procedure [61] has been defined that utilizes the simple conformance test setup of PFC and CN. As with all test procedures, the standard defined requirements (SHALL, SHALL NOT, etc) are extracted and tested via automated tools emulating a link partner, to exercise as many positive and negative test cases as is reasonable.

Some of the most interesting tests involve the testing of what is called the *willing bit* for each TLV. DCBX is broken up into asymmetric and symmetric parameter passing procedures. Asymmetric parameter passing does not require agreement on both sides of the connection, so the *willing bit* is the only field considered when determining if a device should accept a recommendation. Symmetrical parameter passing requires agreement by both peers. In cases where both sides are configured differently but both are willing, the lowest MAC address of the peers is considered to “break the tie”. Many devices have been found to fail all or part of these tests. Another test has shown issues with devices accepting multiple peer scenarios. This generally causes many issues since it is not clear what DCBX frame to follow when a device receives frames from multiple peers. This scenario is commonly seen when a switch improperly forwards DCBX frames, as is the case with some non-compliant devices.

3.3 Applications and Interoperability

The previous sections have addressed the testing needs for conformance. In order to test interoperability of DCB it is beneficial and more informative to observe DCB performance while applications are running on the network. The main application driving the development of DCB is storage.

The iSCSI over DCB interoperability test procedure is broken up into two phases [61]. The initial screening phase utilizes a simple topology. Each converged network adapter (CNA) is connected to a single switch and an iSCSI over DCB target. In this phase devices are tested in a simple topology in order to ferret out any glaring interoperability problems before inclusion in the larger fabric later.

The second phase of testing utilizes all of the devices that passed the screening phase and devices are configured as shown in Figure 3-4, utilizing 3 Initiators, 3 Switches (A,B,C) and 3 Targets at a time. In the second phase of testing, device interoperability is exercised via a series of iSCSI operations and following each test run the switches are moved in a simple rotation such that all of the devices see each other, specifically switches A, B, C are then tested with switch B taking switch A's position, switch C taking switch B's position, etc such that A,B,C is tested, B,C,A is tested, and C,A,B is tested. It is important to rotate the switches so that each initiator and target is directly connected to each switch at least once.

As discussed before, DCBX interoperability causes the majority of network connectivity problems in a DCB network. By causing devices to generate and respond to PFC, the interoperability of PFC can be observed. Even if both devices pass all compliance tests within the accuracy of the observations, it is possible that the timing of PFC generation and response are at the margins and still allow frame loss to occur. ETS interoperability is tested on the full fabric by injecting non-iSCSI traffic into the network on different priorities than assigned to iSCSI. Since the margins for ETS are ±10% of the configured bandwidth value when multiple devices are in a line significant bandwidth deviations can occur.

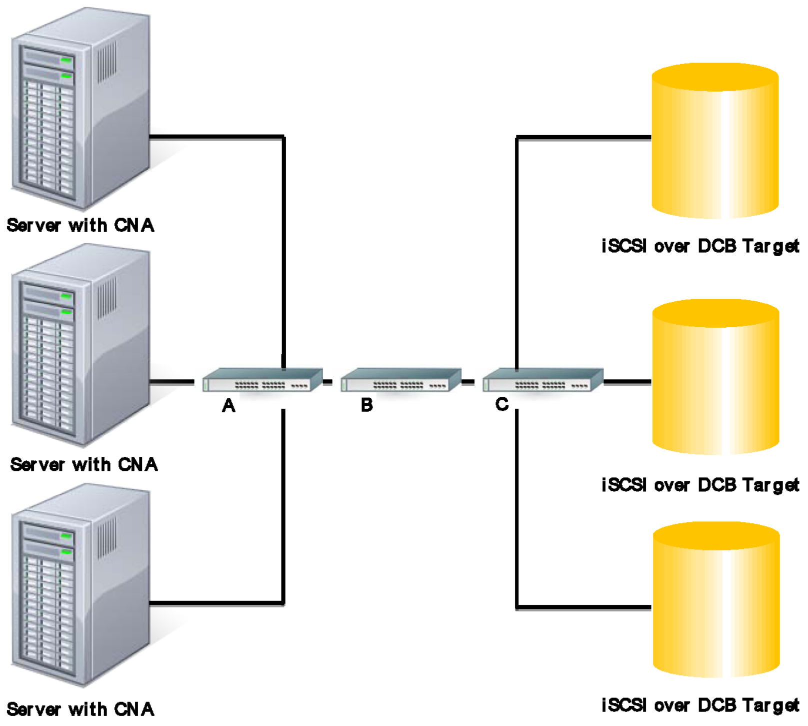


Figure 3-4: Interoperability test setup

One of the biggest challenges in interoperability testing involves distributed monitoring. It is important to monitor each link within a network in order to understand where a protocol error may be occurring. Often times in a large interoperability test setup there might be a problem between only two devices out of nine or ten in the network. Each link needs to be actively monitored during startup to ensure that all of the devices negotiate DCBX properly. The network also needs to be monitored during high load to ensure that PFCs are cascading through the network properly and ETS is throttling the traffic classed bandwidth properly. This can become both extremely costly to obtain that many network analyzers as well as a coordination challenge in order to obtain and follow all of those traces.

Aside from on-demand interoperability testing at UNH-IOL, hosting industry-wide test events also plays a large role in validating early implementations. The Fibre Channel Industry Association (FCIA) and the Ethernet Alliance (EA) both sponsor test events at UNH-IOL. The UNH-IOL also sponsors some events themselves based on demand within the industry. Since September 2008, between the three organizations the UNH-IOL has hosted about four events a year for a total of thirteen events relating to DCB interoperability.

Figure 3-5 shows the number of devices that have participated at each event. The number of participating devices ranged from six to 35.

Figure 3-6 shows the number of interoperability issues that were seen during each event. Each event defined a test plan that include a number of “Test Tracks”. A test track is simply a procedure that is run with a given topology. The number of issues in the graph is determined by counting each step within a test track for a given topology that could not complete with the expected behavior.

Many of the issues identified in the previous sections were also seen during the events as well. Due to the fast paced nature of a test event, extensive debugging is not generally done but there is a focus on identifying issues with the standards. One issue identified during an event is that the T11 organization requires FCoE to be transmitted over a lossless fabric, and if it is not a lossless fabric then the FCoE traffic should not be transmitted. Historically, the IEEE has been a “best effort” fabric, so even if it cannot guarantee a lossless fabric, it will still transmit traffic

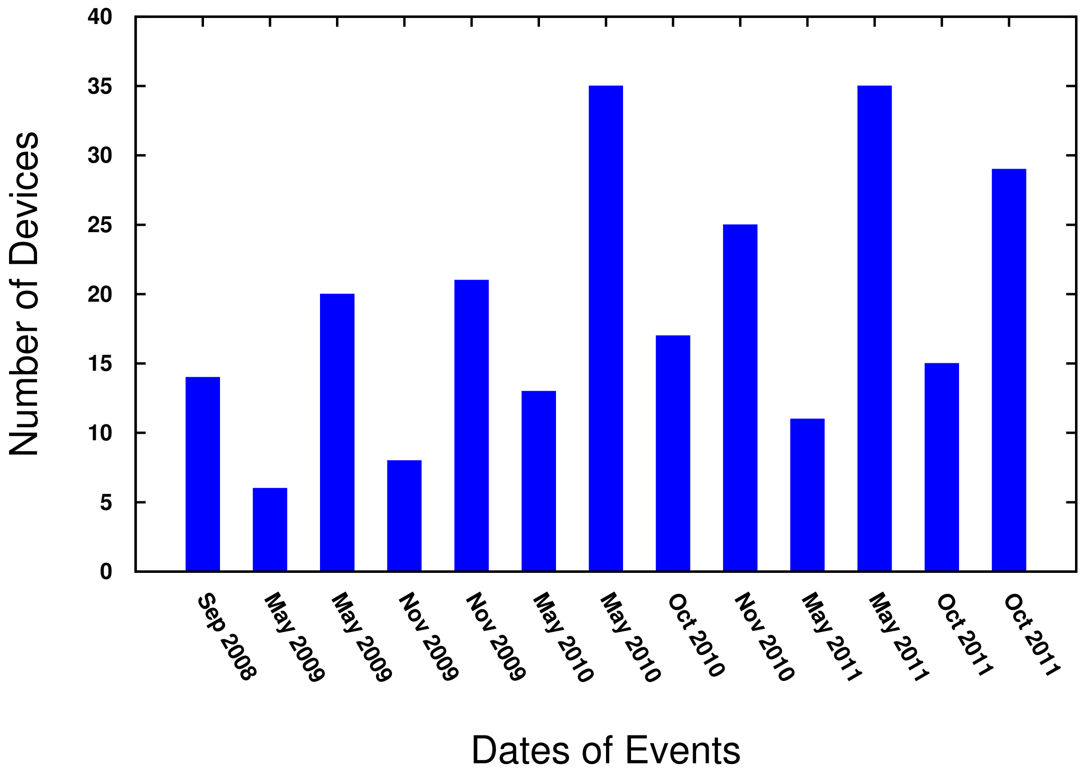


Figure 3-5: Number of devices participating in UNH-IOL test events

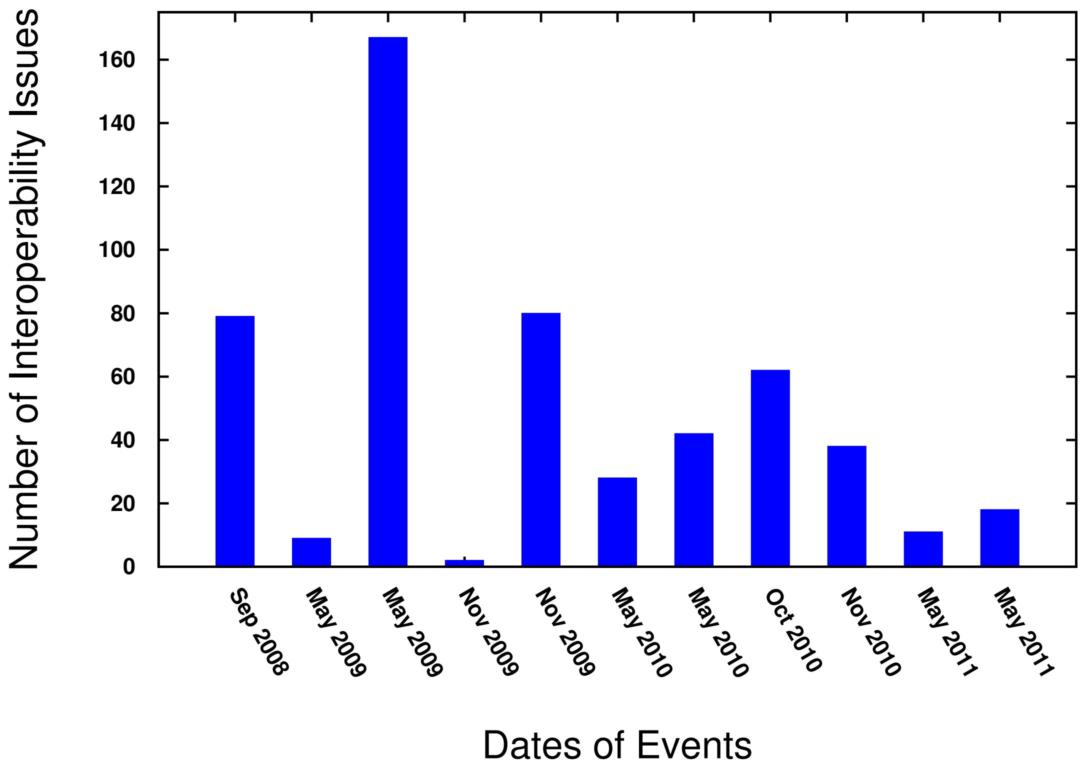


Figure 3-6: Number of interoperability issues identified during UNH-IOL test events

including FCoE. This creates a major implementation difference between vendors implementing T11 and IEEE compliant devices. Another issue seen at an event is in the situation where there is one core switch as the master switch which should be unwilling, the two edge switches should be willing on the port connected to the core switch but unwilling toward the end nodes. The plan was for the core switch to tell the edge switches how to configure and the edge switches to push that down to the end nodes, but there is no Indication in the standard how the edge switches are to take the configuration from the uplink and push it to the down link.

3.4 Conclusion

Interoperability and Conformance between various manufacturer's iSCSI over DCB devices is now more important than ever in this nascent space to ensure the industry's collective goals of growth and adoption. An iSCSI over DCB Integrator's List is a co-operative effort between companies that are dedicated to implementing the IEEE and IETF standards for DCB and iSCSI and to promote interoperability between devices from different manufacturers and facilitate industry adoption of the technology. An Integrator's List helps ensure high-quality iSCSI over DCB products are recognized in the marketplace, while simultaneously offering a clearing house of network testing ideas and best known practices to be centralized from the Industry, thereby enabling a first hurdle of acceptance before purchasers further investigate the pros and cons of distinct products.

Without such a program, two approaches, neither of them optimal, tend to dominate interoperability efforts. In the lesser of these two evils, vendors end up spending an exorbitant amount of money to establish and maintain their own interoperability facilities, where the focus is more on their product's interoperability rather than standard's based conformance and interoperability of same. A worse-case scenario occurs when vendors simply do not perform interoperability testing and hope that the first (smaller) group of vendors will do the testing against their devices. iSCSI over DCB interoperability has been improving due to large efforts of the first group of vendors, despite the high price of pursuing this solution.

Many enterprise customers still see iSCSI over DCB as an unfamiliar and unproven technology. While implementers have been working aggressively with UNH-IOL to develop rigorous standards based conformance and interoperability testing, their efforts must become known for confidence to be built. To this end, the voluntary iSCSI over DCB Integrator's List is seen as a vehicle to help promote the network testing efforts of those participating [62].

Looking ahead there continues to be testing and deployment challenges for this new technology. As more vendors begin to enter the market, it is important to develop a strong testing program that incorporates both conformance and interoperability. The first wave of early adopter vendors has already been participating in testing programs to ensure that they work with each other. Once the second wave of adopters enters the market it will be very important to test and ensure that these new implementations interoperate with the early adopters.

Chapter 4

iSCSI on a Converged Data Center Network

The previous chapter provided a detailed examination of each protocol within DCB and some of the challenges involved in trying to measure and test the protocols in real-world devices. This chapter evaluates the effects of converged traffic classes on a data center grade network. We examine various traffic streams sharing a modern data center network prior to receiving actual DCB hardware. Finally this chapter ends with an analysis of how future DCB enabled hardware will benefit a converged network.

4.1 Introduction

In order to evaluate the pros and cons of the DCB protocols, it is important to first understand how the performance of various traffic flows are impacted by a converged network. However, there are very few papers that analyze traffic flows on converged networks since this is a relatively new area and standards are still being defined. Ranganathan and Subhlok [63] analyzed the cost and benefits of a converged network over a backplane. They performed a broad analysis including cost of each port, cost of different interfaces and performance implications of their choices. The paper examined the implications of different lane widths and PCI Express speeds. The study concluded that a converged network could reduce cost significantly. They found that larger fabrics offered the most performance and cost savings only if they are fully utilized, while smaller fabrics offered better fabric isolation and finer grain scalability.

There are no papers that evaluate the performance of LAN and SAN traffic on a converged data center network. In this chapter, we examine the performance of LAN-TCP/UDP and SAN-iSCSI traffic on a converged network. The LAN and SAN traffic flows have very different characteristics. iSCSI encapsulates SCSI which has a request response behavior. In the case of a Read, a single request results in a large amount of response data. In the case of a Write, a large number of requests will likely be followed by a single response indicating that the operation completed successfully. TCP is a reliable network transport protocol, which means that for every frame an acknowledgment will need to be sent back to indicate success, while UDP is unreliable, which means that there is no returning acknowledgments.

There are no papers that evaluate the performance of iSCSI/TCP/UDP over a converged network. Most recent iSCSI performance papers either focus on comparisons to Fibre Channel [53, 64, 65] or they examine system level improvements such as caching and protocol tweaking [52, 66]. We analyze the impact of congestion when a network is shared by TCP, UDP, and iSCSI traffic. We use data center grade equipment and create multiple virtual networks designed to carry iSCSI SAN and non-storage LAN traffic. By creating the converged network we are able to measure the effect of loss and other traffic on the performance of the LAN and SAN traffic. Using multiple, separate virtual networks we are able to isolate the traffic classes which provides a basic level of security that Data Centers today demand.

We evaluate the impact of congestion on a converged network as the number of traffic classes increases and traffic types differ. The devices in the converged network are of five types: LAN Source, SAN Source, LAN Sink, SAN Sink and a Switch. The SAN Sources submit I/O requests to the SAN Sinks, the LAN Sources submit TCP and UDP requests to the LAN Sinks, while the switch transmits data between all of the devices. The sources and sinks generate traffic over the same network, thereby, causing congestion.

4.2 Experimental Setup

The experimental setup is shown in Figure 4-1. The server platforms are powerful multi-core systems with more than 2GB of memory to ensure that the traffic classes used in this study can utilize a large portion of the network. The Ethernet adapters are powerful 10 Gigabit Ethernet adapters from leading next generation adapter manufacturers Chelsio and Intel. The switch is a 10 Gigabit cut-through switch from Fujitsu that provides near wire speed transmission on the network, which ultimately provides near wire rates of throughput and latency. The iSCSI Initiator and Target is the open source software provide by the University of New Hampshire InterOperability Laboratory.

The application software we use in this study is the dd and netperf applications run in a Red Hat Linux Operating System. The dd application is used to exercise SAN traffic and the netperf application is used to exercise the LAN traffic class. The dd application is a simple SCSI I/O application that allows the user to specify whether to do a read or write, the number of operations and the size of each operation. The netperf application is an industry recognized benchmark tool that uses a client and server model to exercise several different traffic patterns such as one-way TCP, one-way UDP, two-way TCP and two-way UDP.

Our experimental SAN workload consists of 2000 one megabyte read and write

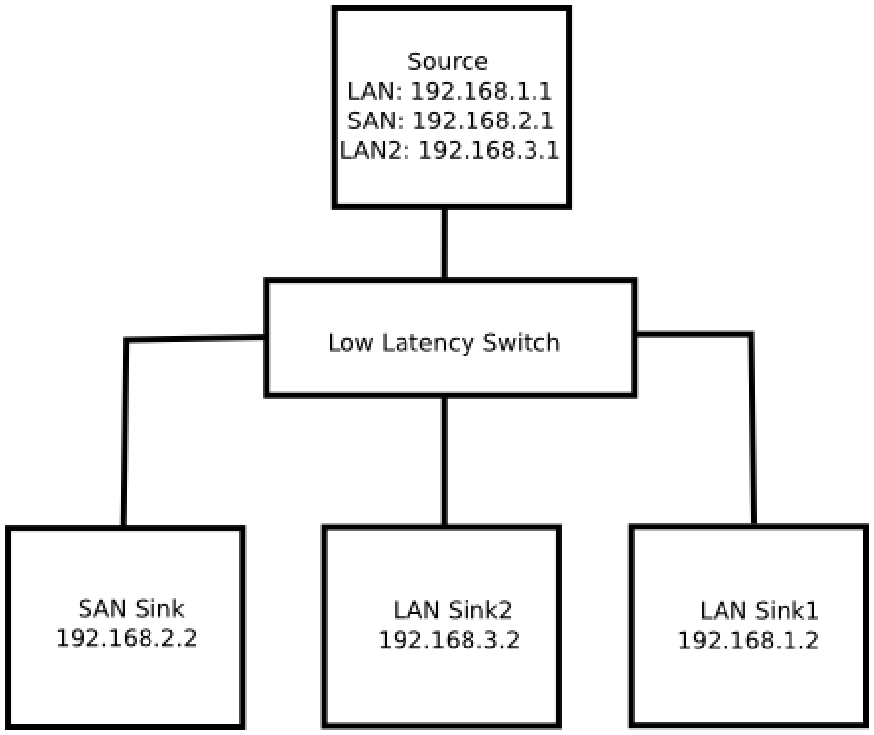


Figure 4-1: Experimental setup

operations. The parameters provided to the dd application to create this workload consist of count=2000 and bs=1048576. The 2000 count ensures that each experiment runs long enough to be stable, while the megabyte requests in the experimental workload ensures a large load on the network. Furthermore, this is the largest SCSI workload that can be handled by the network without overrunning the capacity of the target. We use the memory I/O mode of the iSCSI Target since it keeps the read and write operations in memory and does not send traffic to the disks. This allows the iSCSI Target to provide the most performance available because we want to focus on the network and not the disk drives.

Our experimental LAN workload consists of one-way TCP and one-way UDP transactions. The parameters provided to the netperf application to create this workload are "-t TCP_STREAM" and "-t UDP_STREAM". The converged network is created by setting up three virtual Ethernet interfaces over the Chelsio adapter in the source system on different subnets using the command: ifconfig eth2:1 <ip-address> <netmask>. The UDP workload is an example of a typical streaming video or audio, and the TCP workload is an example of web, email, and socket applications.

We first measure baseline performance metrics by running solo traffic classes on the network. Next, all workloads are submitted to the network in parallel and performance on the converged network is measured. The performance metric analyzed in our experiments is the throughput of each workload.

4.3 Two Traffic Classes

During the first set of experiments, the SAN Sink and LAN Sink1 (Figure 4-1) are used. Figure 4-2, Figure 4-3, Figure 4-4, and Figure 4-5 show the results of our experiments. The LAN traffic on the network varies between TCP and UDP, while the SAN traffic varies between ISCSI read and write. Each graph plots the throughput of a traffic class when the network is not shared and when the network is shared. The lines show how throughput of a traffic class varies when the network is not shared versus when the network is shared.

Our TCP graphs show that the TCP traffic has higher priority than SAN traffic since the throughput of both Reads and Writes dropped in the converged network while the throughput of the TCP traffic remains unchanged. Comparing the TCP Read and TCP Write graphs, the Read throughput drops less than the Write throughput. Our UDP graphs show that the UDP traffic does not seem to have higher priority than SAN traffic. In the UDP and iSCSI Read graph, the SAN throughput is reduced minimally while UDP throughput decreases significantly on the converged network. In the UDP and iSCSI Write graph, the SAN line decreases about the same as the LAN line.

These results show that the type of traffic has a direct effect on the performance

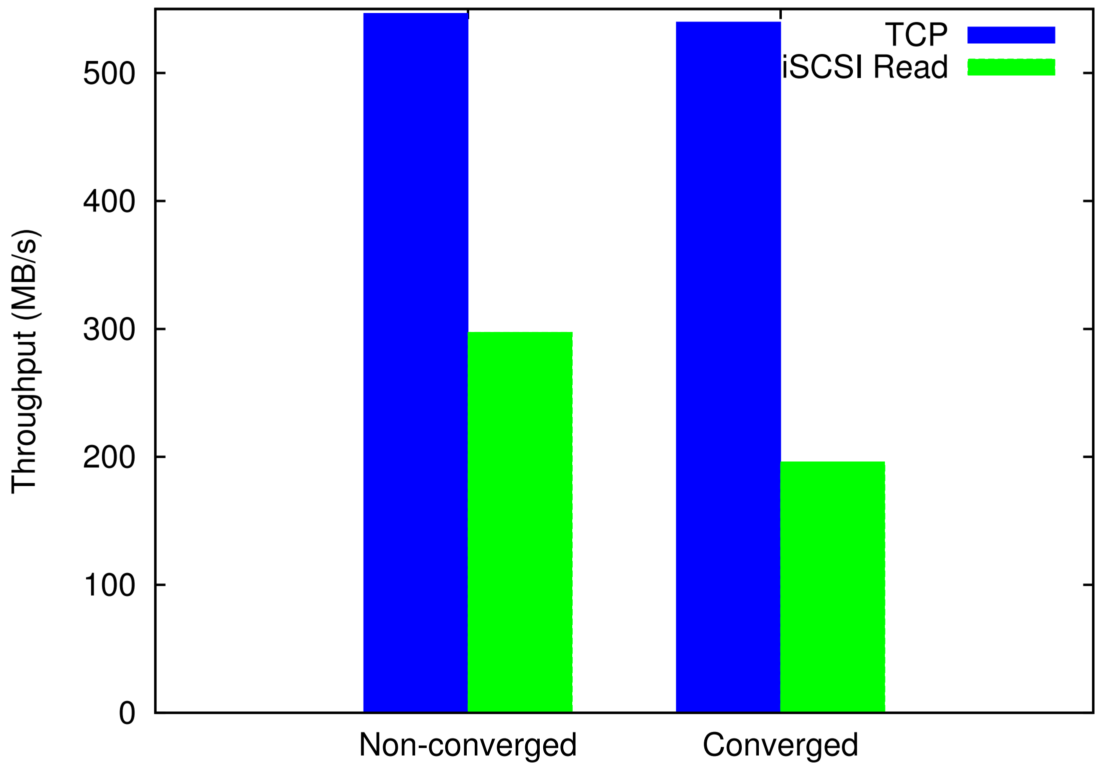


Figure 4-2: Throughput of TCP converged with iSCSI Read

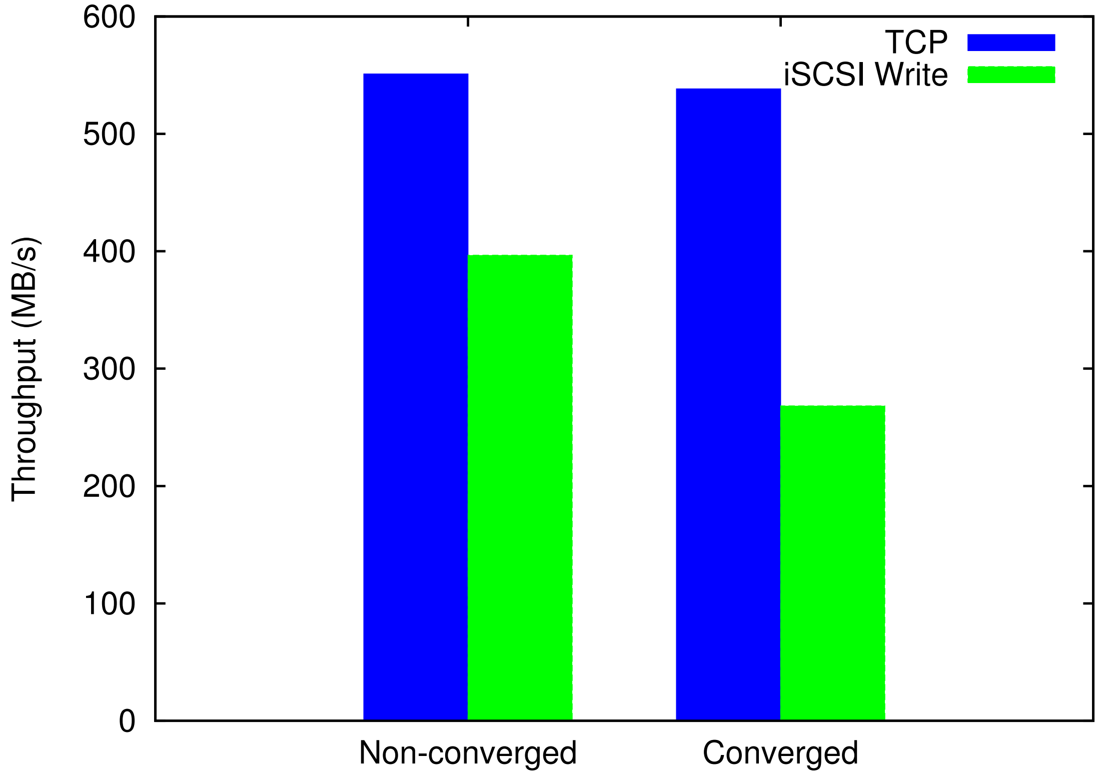


Figure 4-3: Throughput of TCP converged with iSCSI Write

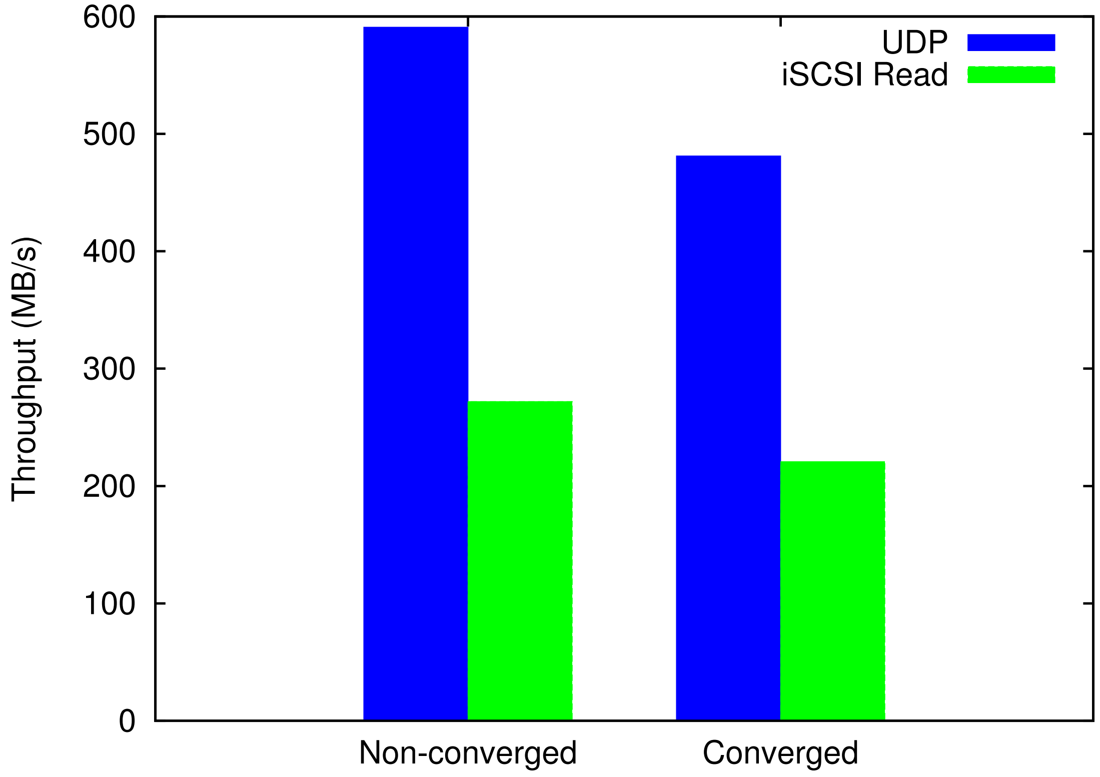


Figure 4-4: Throughput of UDP converged with iSCSI Read

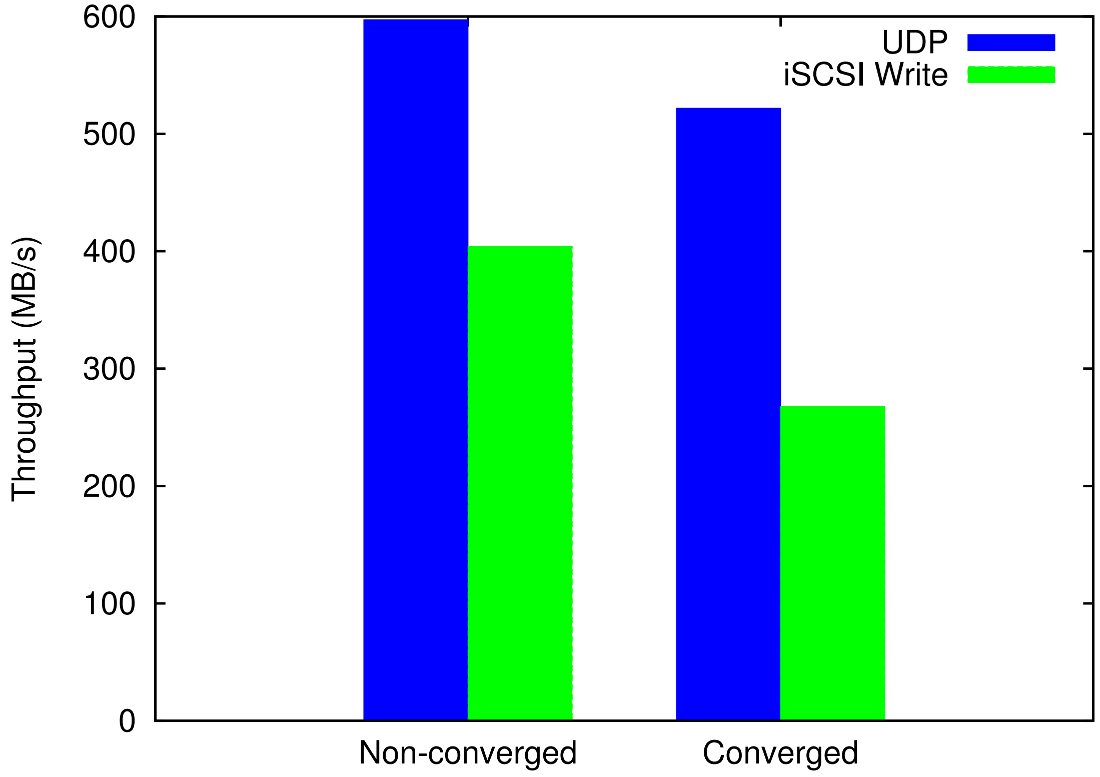


Figure 4-5: Throughput of UDP converged with iSCSI Write

of the traffic classes on a converged network. The UDP and the iSCSI Writes both have a one-way traffic pattern, while the TCP and the iSCSI Reads have a two-way traffic pattern. Our results show that two-way traffic patterns have a higher priority than the one-way traffic patterns on a converged network. Our experiments also show that similar traffic classes perform similarly.

4.4 Three Traffic Classes

During the three traffic class experiments, all of the devices in Figure 4-1 are used. In this experiment we vary the traffic between TCP, UDP and Read, Write operations. Figure 4-6 and Figure 4-7 show the effect of a converged network on the throughput of three traffic classes. Each graph plots the throughput of various traffic types when the network is non-converged and converged. The left column of each graph shows the throughput of each traffic class while running on the network by itself. The right column of each graph shows the throughput of all three traffic classes while running on the network simultaneously.

The Three Traffic Class experiment confirms the results from the Two Traffic Class Experiment. Figure 4-6 shows that the TCP throughput decreases significantly less than the UDP and iSCSI Write throughput in the converged network. Figure 4-7 shows that the iSCSI Read throughput decreases far less than the throughput of the other traffic classes. The TCP throughput decreases less than the UDP throughput, but TCP performance falls far more than iSCSI Read. We believe that the TCP throughput decreased because of increased congestion in the three-traffic experiments compared to the two-traffic experiments. The congestion on the network impacted TCP traffic more than the iSCSI traffic because iSCSI Read traffic only sends small request packets from the source to the SAN Sink, the network flow direction with major congestion. The major traffic generated by iSCSI is the large read packets

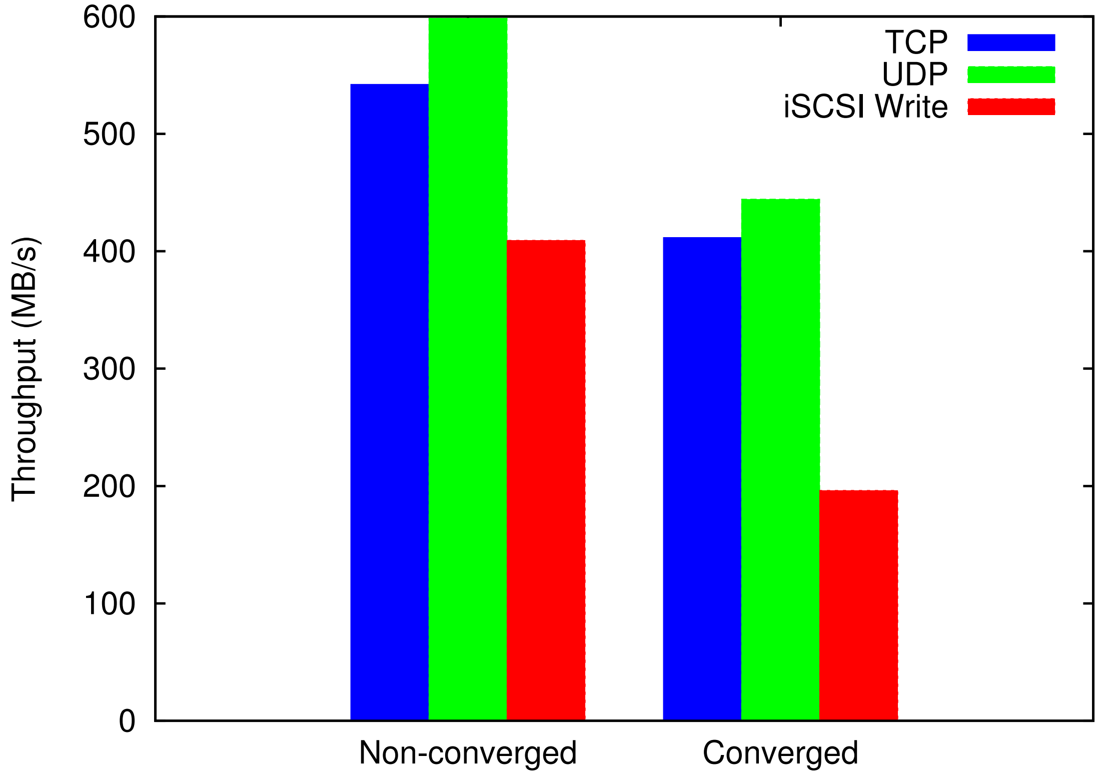


Figure 4-6: Throughput of TCP and UDP converged with iSCSI writes

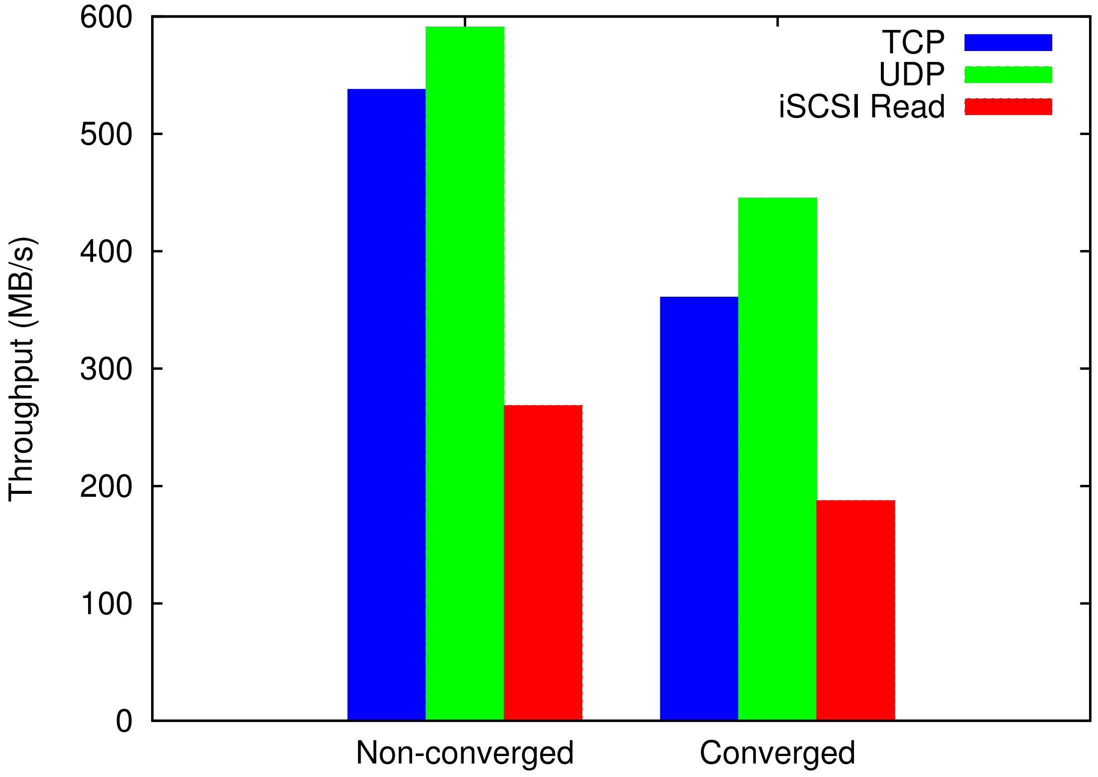


Figure 4-7: Throughput of TCP and UDP converged with iSCSI reads

from the SAN Sink to the Source, the network flow direction with less congestion.

4.5 Analysis

Our results show that iSCSI Write traffic performance is severely degraded when it shares the network with TCP and UDP traffic streams. This problem can be addressed using the ETS protocol which can be used to shape the bandwidth of an egress port of a device. In this case, the most benefit would be seen on the egress port of the Traffic Source. The traffic classes can be tagged with priorities such as TCP=1, UDP=3, SAN=6 and placed in their own Priority Group ID. This setup allows the bandwidth to be configured to ensure that SAN traffic gets more guaranteed bandwidth. In the three traffic classes experiments, the iSCSI Write received less than 25% of the total bandwidth. The ETS technique can ensure that iSCSI Write receives a greater percentage of the total available bandwidth.

The PFC protocol may not be useful for the experimental platform in this chapter. However, consider a new experimental platform with a single server node (sink) that provides several services and multiple clients (sources) connect to this server and use its services. In this setup, the multiple clients could overload the link between the switch and the new sink. This problem could be addressed using the PFC protocol which can pause the lower priority TCP and UDP traffic classes to ensure that the higher priority iSCSI traffic continues unimpeded.

If another switch is added to the new platform, and more clients connect to this switch, then a situation may arise where the CN protocol would be useful. Without CN, the first switch could use PFC to pause traffic classes as the link between the switch and sink gets overloaded. When the first switch pauses traffic, the second switch could get overloaded resulting in the second switch using PFC to pause traffic. This lowers the performance of high priority traffic such as iSCSI and UDP within a priority that were not necessarily causing the original congestion. The CN technique could allow the overloaded link to preemptively slow some of the traffic streams and prevent PFC from being issued.

4.6 Conclusion

As the industry moves toward converged networks, performance analysis of different traffic classes and protocols combined on a single fabric becomes important. Using a Data Center grade 10 Gigabit Ethernet network, this study addresses the performance impact of two and three traffic classes sharing the same network. We found that the impact of congestion varies with each traffic class. The two-way traffic classes were less impacted by congestion than the one-way traffic classes. The two-way traffic classes utilize the network more efficiently, especially in the case of the iSCSI Read because most of the traffic returns on the less congested side of the network. Our study suggests how the protocols defined by DCB could be used to mitigate the impact of congestion. For example, this study shows that iSCSI Write traffic is severely effected by a converged network. The ETS protocol could be used to guarantee a minimum bandwidth percentage so that the higher priority SCSI traffic does not slow down precipitously. In the worst case, PFC protocol could be used to halt the lower priority TCP traffic and ensure that the mission critical UDP and SAN traffic is not slowed down. If the fabric significantly more complex than in this study, the CN protocol could be used to avoid pausing traffic in unrelated streams.

Future work could explore larger and more complex fabrics. We limited the number of traffic classes due to a lack of access to more hardware. In our platform, each traffic class needed its own server hardware, which limited the extent of the testing. Future studies could look at more traffic classes with more switches in the fabric. These studies can help highlight key problems with congestion in converged networks and their impact on the various traffic classes. Since the DCB standards are defined to solve congestion problems in converged networks, the results of these experimental studies would be useful to the DCB task group.

Chapter 5

Performance Evaluation of DCB's Priority-based Flow Control

The previous chapter provided a baseline analysis of the performance of different application's on data center grade hardware without DCB enabled. This chapter builds upon that base by utilizing DCB enabled hardware to provide an analysis of the performance impact on latency and throughput of two common applications: storage and inter-processor communication. For storage, we examine the effect that enabling PFC on a network has on the throughput of iSCSI Read and Write operations. For inter-processor communications, we observe the effect of enabling PFC on the latency of MPI applications.

5.1 Introduction

We evaluate the impact congestion has on traffic streams within a single priority in a converged network. DCB allows a network to be broken up into eight different traffic classes. In data centers there are more than eight applications that will require lossless behavior; therefore, we are interested in seeing how applications sharing a traffic class will perform. We experimentally measure the effect of enabling and disabling PFC on the throughput of a traffic stream. The experimental setup is shown in Figure 5-1. The server platforms are powerful multi-core systems with more than 2GB of memory to ensure the traffic classes used in this study can utilize a large portion of the network. The Ethernet adapters are dual port X520 Converged Network Adapters (CNA) with the 82599 processor from a leading next generation adapter manufacturer, Intel. The switch is a 10 Gigabit cut-through switch from Fulcrum that provides near wire speed transmission on the network. New hardware is important because DCB requires significant changes in order to support the technology. Multiple transmit and receive queues are required in order to support independently pausing different priorities.

The topology shown in Figure 5-1 was chosen because we believe it is representative of any current network regardless of its complexity. Current Ethernet networks use some variation of Spanning Tree Protocol (STP) in order to route traffic through the network. Loops in a network can cause traffic to be routed indefinitely within the network and never reach its destination. STP was defined in order to prevent loops in the network from occurring. STP builds a complete node tree of the network with a single root and a connection to every node in the network. STP successfully prevents loops and provides adequate performance for standards compliant Ethernet networks of less than eight hops in a line. In this case, a hop is equivalent to a switch on the network.

If the tree formed by STP is broken up into smaller units, it can be observed that every communication between two end points will be going through a common link similar to that found in our topology. Near the root, congestion will be especially high for traffic being routed from one side of the tree to the other. Thus, regardless of the size of a network, if it is using STP the network can be reduced down to a series of topologies as seen in Figure 5-1. The Multiple Spanning Tree Protocol (MSTP) was defined in order to limit this congestion scenario. MSTP allows different VLANs to define their own spanning tree thus reducing the congestion between VLANs. Even with MSTP, complex networks will still reduce down to this common link topology on a per VLAN basis.

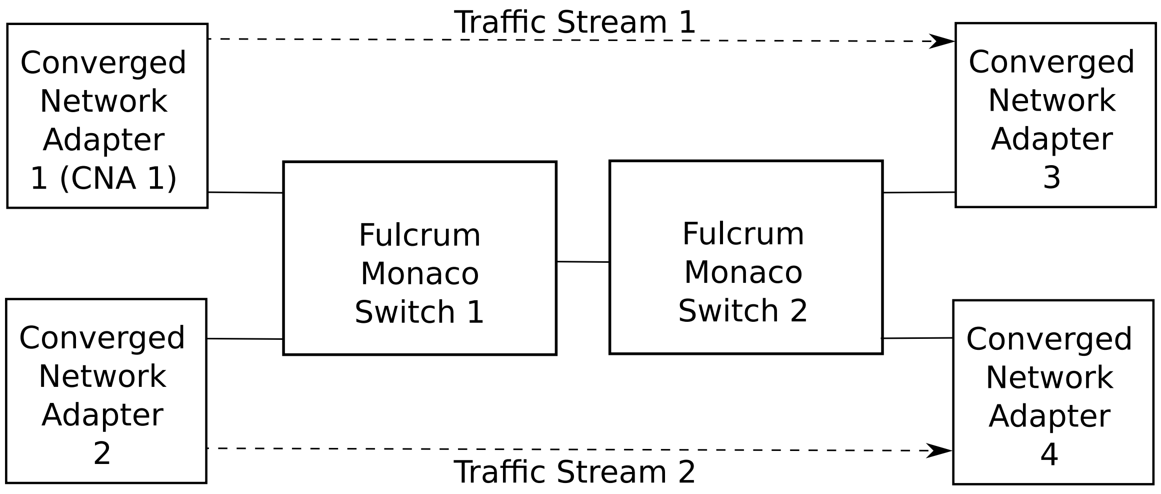

Traffic Stream 1
Converged Network Adapter 1 (CNA 1)
Converged Network Adapter 3
Fulcrum Monaco Switch 1
Fulcrum Monaco Switch 2
Converged Network Adapter 2
Converged Network Adapter 4
Traffic Stream 2

Figure 5-1: Experimental setup

The application software used in this study is the UNH-IOL developed blaster program run in an openSUSE Linux Operating System. The blaster program is a simple command line application utilizing raw Ethernet sockets to transmit user defined frames on an Ethernet network in a multi-threaded manner. A raw Ethernet socket is a low-level socket with syntax similar to a standard TCP socket. Unlike a TCP Socket, a raw Ethernet socket allows a programmer to define the frame characteristics all the way down to the Ethernet MAC (Media Access Control) layer. The blaster program allows users to define a frame in a simple plain text file. It also permits users to specify the number of threads to use, the delay between frame transmissions and total number of frames to transmit.

For the MPI results, we use the Open MPI implementation of MPI [67]. To measure the latency of the system, we use the Intel MPI Benchmarks [68]. All four systems are used to measure the performance of MPI with each system having two processor cores for a total of eight cores.

Our experimental traffic workload consists of 1000-byte unicast UDP frames with the MAC address fields defined to transmit frames from CNA 1 to CNA 3 and from CNA 2 to CNA 4 as shown in Figure 5-1. The minimum standard Ethernet frame size is 64 bytes and the maximum is 1518 bytes. The frame size of 1000 bytes serves to demonstrate the performance implications on a medium sized standard Ethernet frame. To generate the normal high bandwidth results, the parameters passed to the blaster program are: 0 delay, 4 thread and unlimited run time. To generate the low bandwidth results, parameters passed to the blaster program are: .001 second delay, 16 threads and unlimited run time.

The netstat program is used to record results. Netstat is a simple network monitoring tool available in most Linux operating systems. The command line parameter of '-ic' is used in order to continuously list the throughput from all of the interfaces in the system. This command reports the throughput numbers once a second. About 300 samples of each experimental run are recorded.

We first measure baseline performance metrics by disabling PFC and running the blaster program with different values for the number of threads. In the later experiments, all workloads are submitted to the network in parallel and the performance is measured. The performance metric analyzed in our experiments is the throughput of each workload as measured from CNA 1.

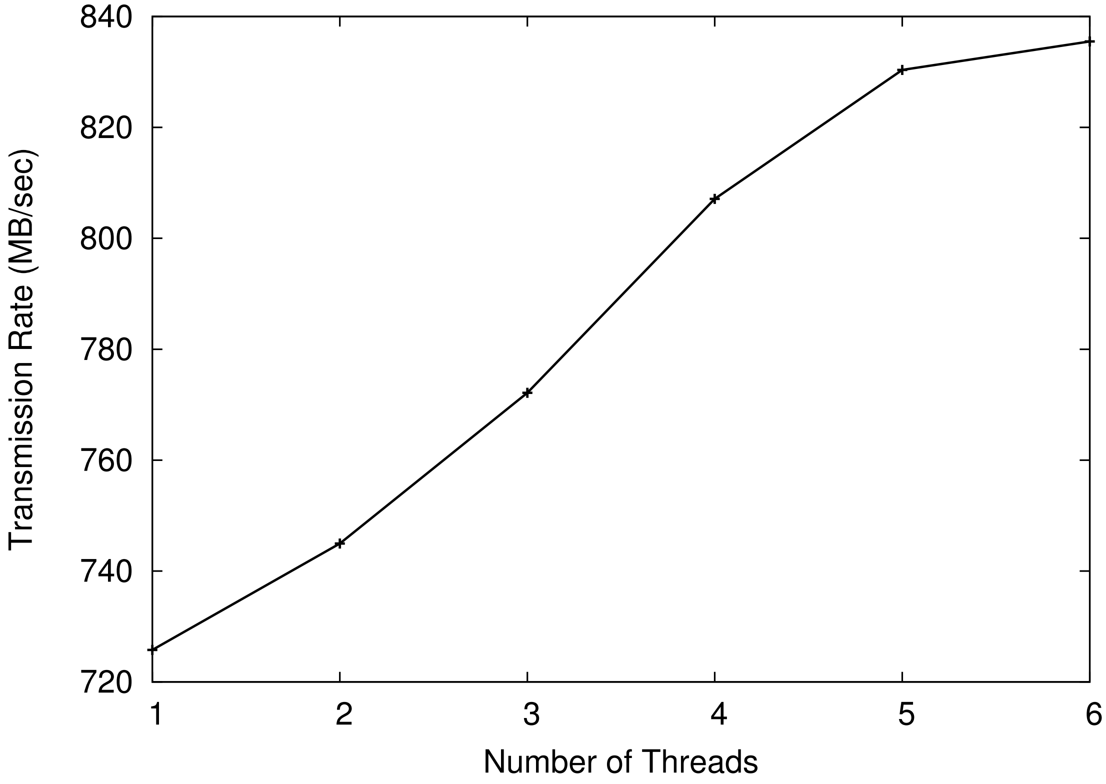


Figure 5-2: Throughput as the number of threads increases

5.2 Number of Threads

During the first set of experiments, the CNA 1 and CNA 3 devices (Figure 5-1) are used. Figure 5-2 shows the results of our experiments. PFC is disabled on both CNA 1 and CNA 3 and the custom UDP traffic is sent from CNA 1 to CNA 3 via the blaster program. The number of threads are varied from 1 to 6 and the throughput is measured.

The graph shows as the number of threads increases, the performance levels increase linearly. The throughput scales from about 700 MB/s to over 800 MB/s. At four threads the throughput approaches the limits of 10 Gigabit Ethernet and as the number of threads increases further, the limits of 10 Gigabit Ethernet and context switching between the large number of processes causes the throughput to level off around 835 MB/s.

These results show that the blaster program is capable of driving enough traffic to create an adequate level of throughput on the network. Additionally, these results show four threads saturates the network. This is very important for later experiments so that we are confident lower throughput numbers are the result of DCB mechanisms and not a limitation of our traffic generation software. After four threads the throughput does not increase due to reaching the limits of 10 Gigabit Ethernet and context switching between too many threads. For the rest of the chapter, 4 threads are used to generate frames, unless otherwise noted.

5.3 PFC versus No-PFC

During the PFC versus No-PFC experiments, the CNA 1 and CNA 3 devices are used. We turn PFC on and off on both CNAs and measure the throughput of the blaster program running the UDP traffic workload. Figure 5-3 shows the effect of turning PFC on and off on the throughput of the blaster program. The left column shows the

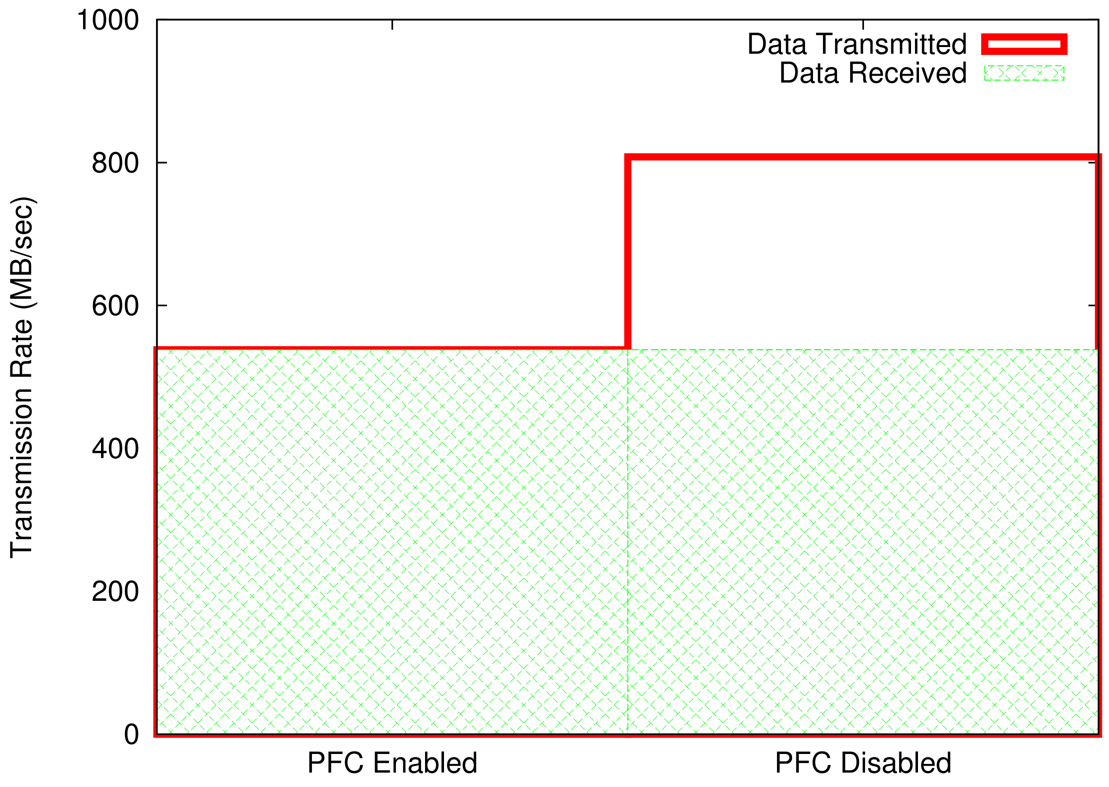


Figure 5-3: Throughput when PFC is enabled and when PFC is disabled

throughput of the blaster program with PFC turned on and the right column shows the throughput of the blaster program with PFC turned off. During both tests the number of threads specified on the blaster command line was four.

The PFC versus No-PFC experiment confirms results from the previous experiment that four threads are sufficient because the receiver cannot handle more than 538 MB/s. Figure 5-3 shows that with PFC turned on, the throughput is reduced from over 800 MB/s to just over 500 MB/s, showing that CNA 3 is the bottleneck in the system. As CNA 3 becomes overwhelmed with processing the incoming traffic it begins to send PFC frames to pause the traffic. The PFCs cascade back through the network from CNA 3 to the originator of the traffic, CNA 1.

With PFC on, no frames were lost during the experiment. With PFC turned off, CNA 1 transmitted a total of 429 GB of data during the observation period and CNA 3 received 260.9 GB of data. The total amount of data lost during the observation period was 168.1 GB or about 40%.

The results of the PFC versus No PFC experiment confirmed the results of the threads experiment and showed that the end nodes were the bottleneck in the system. As the end node became congested trying to process the incoming traffic, it began to transmit PFC frames to the switch it was directly connected to. The PFC frames then cascaded backwards through the network to the traffic originator and reduced the output of the originator from about 800 MB/s to 500 MB/s. This confirms in a simple traffic source to traffic sink system that PFC works as expected.

5.4 Two Full Throughput Devices

During the Two Full Throughput Devices experiments, all of the devices in Figure 5-1 are used. In this experiment, PFC is enabled on all of the devices in the network. The throughput is first measured between CNA 1 and CNA 3 running by itself, then

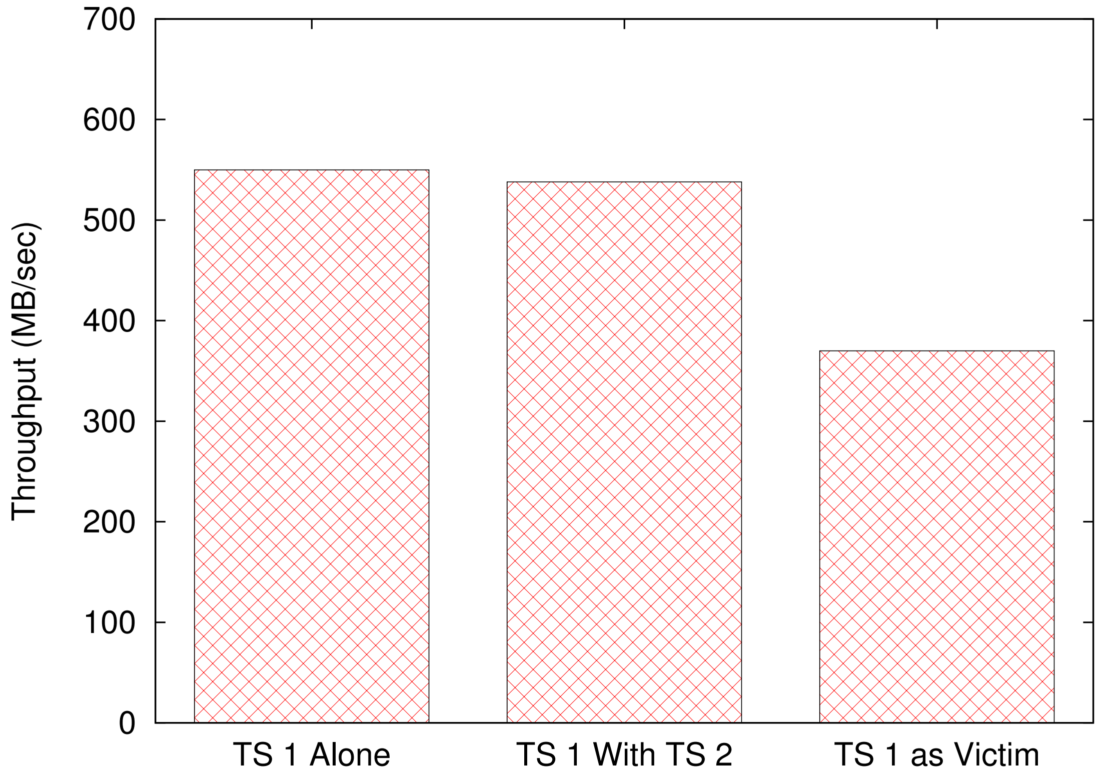


Figure 5-4: Two full throughput devices: throughput of CNA 1

traffic between CNA 2 and CNA 4 is started. The Victim point, which is a term that comes from the IEEE working group [69], shows what happens to the throughput between CNA 1 and CNA 3 when the throughput is temporarily reduced between CNA 2 and CNA 4. Figure 5-4 shows the effect starting another traffic stream has on the original traffic stream between CNA 1 and CNA 3.

The Two Full Throughput Devices experiments show the throughput of the streams is minimally affected by adding another stream. The throughput is reduced from about 550 MB/s (TS 1 Alone) to about 530 MB/s (TS 1 With TS 2). This shows both streams are able to completely utilize the theoretical throughput of the 10 Gigabit link between the switches and a simple two stream system can converge to be well balanced using only PFC. Additionally, it shows an excellent level of fairness on the network for both streams as they both received about the same amount of network utilization.

The TS 1 as Victim data point shows the limitations of a network with only PFC enabled and no end-to-end flow control. When the throughput between the other CNAs is temporarily reduced by increasing the number of PFCs coming from the CNA 4 system, it can be seen that the throughput between CNA 1 and CNA 3, which should not be affected, is greatly reduced to less than 400 MB/s. The throughput of the victim stream was nearly cut in half from 550 MB/s to 350 MB/s and the network appeared to converge to the lowest throughput capable device. This shows how PFC meant for one stream impacts other streams with the same priority in a network. With end-to-end flow control on the network, the number of PFC frames on the inter-switch link (ISL) would be limited and the impact on other streams could be reduced.

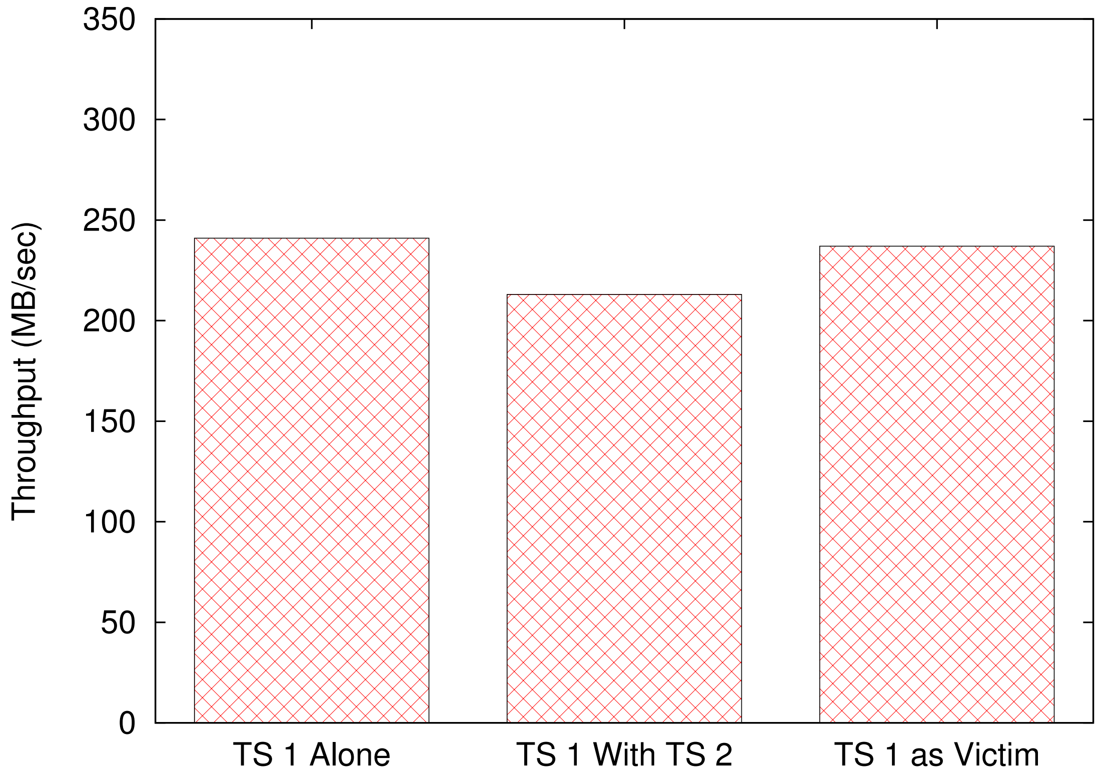


Figure 5-5: Low throughput device on a converged network

5.5 Low Throughput Device on a Converged Network

During the Low Throughput Device on a Converged Network experiments shown in Figure 5-5, all of the devices in Figure 5-1 are used. In this experiment, PFC is enabled all of the devices in the network and the throughput is observed on CNA 1. The throughput is first measured when CNA 1-CNA 3 blaster transmission is running at low bandwidth by itself labeled LowBW in Figure 5-5. Next, full bandwidth blaster traffic between CNA 2 and CNA 4 is started. Finally, the full-bandwidth blaster throughput between CNA 2 and CNA 4 is temporarily reduced by starting a processor intensive application on CNA 4. During the entire experiment the number of threads on CNA 1 is set to 16 along with a 0.001 second delay to create the low throughput load. The number of threads specified on CNA 2 is 4 with no delay to create the high throughput load.

The experiments show that when utilizing 16 threads and a 0.001 second delay, the blaster program on CNA 1-CNA 3 gets a throughput of about 250 MB/s on a network by itself (TS 1 Alone bar in Figure 5-5). When the traffic stream from CNA 2-CNA 4 that supports PFC is added to the network, the throughput is minimally reduced to about 200 MB/s (TS 1 With TS 2 bar in Figure 5-5). This demonstrates that even when two streams are not fully utilizing the full bandwidth of the 10 Gigabit network, the low throughput traffic stream is significantly reduced by the other stream being added. This corroborates the results of previous experiments. Interestingly, when the throughput of the second stream is temporarily reduced and more PFC frames are seen on the network, the throughput of the first stream recovers to about 240 MB/s (TS 1 as Victim bar in Figure 5-5).

The results of the low throughput test are very surprising. When a full throughput traffic stream from CNA 2-CNA 4 is converged with a low throughput stream from CNA 1-CNA 3, the low throughput stream is reduced by about 20% from 250 MB/s to 200 MB/s. The low throughput stream by itself did not generate any PFC frames in the network but as the high throughput link is added, PFCs began to be seen on the link between the two switches from Switch 2 to Switch 1. As the link between the two switches became congested, Switch 1 began to send PFC frames to the traffic sources (CNA 1 and CNA 2). This should have reduced the throughput of the low throughput stream by more than 20%, but Switch 1 began to aggressively send pause off PFC messages to the low throughput source at CNA 1. As discussed earlier the PFC frame is sent with a time value. A PFC frame with a time value of 0 essentially turns pause off and allows a device to begin transmitting immediately. Most implementations tend to just send PFC frames with the maximum pause time specified. Through aggressive use of PFC frames with a time of 0, the switch is able to minimize the throughput reduction to the low throughput stream and allow it to transmit data more than expected. The switch was able to transmit more pause off PFC messages because the ingress queue of the low throughput stream (CNA 1) drained faster than the ingress queue of the high throughput queue (CNA 2).

When the throughput of the high load stream is temporarily reduced, the results are even more surprising. Based on the results of the two high bandwidth traffic streams experiment, the throughput of the low throughput stream was expected to be reduced even further as more PFCs would be seen on the inter-switch link. Instead, the low throughput stream actually recovered some of the lost throughput and returned to just below what it achieves without another stream on the network. This is attributed to even more use of time 0 PFC frames by the switch.

These results demonstrate that as more traffic is converged on the network, PFC begins to show significant limitations. Congestion Notification is designed as an end-to-end flow control mechanism and should alleviate many of the limitations of PFC. Unfortunately, CN is the most complicated technology defined in DCB and does not have much support in real implementations found in the industry.

5.6 Latency of MPI on a Converged Network

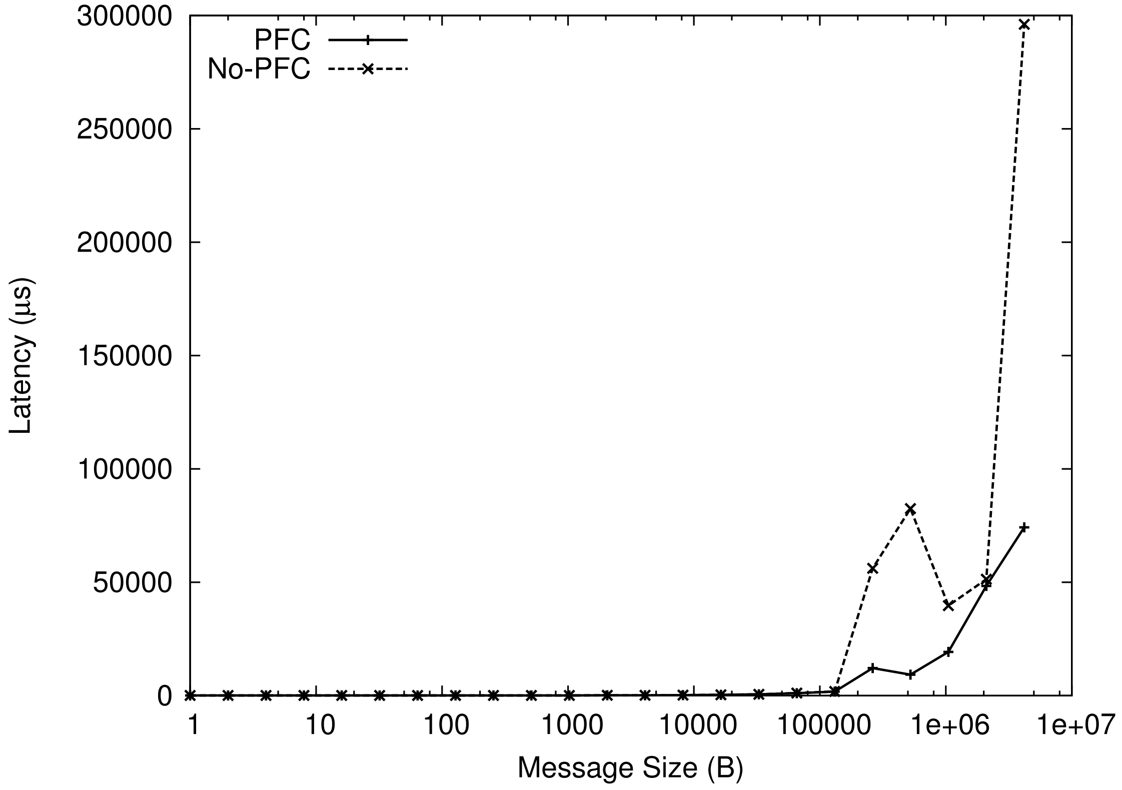


Figure 5-6: MPI exchange benchmark latency

During the Latency of MPI on a Converged Network experiments, all of the devices in Figure 5-1 are used. In this experiment, Open MPI is installed on all systems and Intel MPI Benchmarks are executed on the systems. The latency is first measured with PFC turned off, then the latency is measured again with PFC on. With PFC off, MPI relies on TCP for flow control. The figures are based on the Exchange benchmark with all eight processes in use. This benchmark increases the message

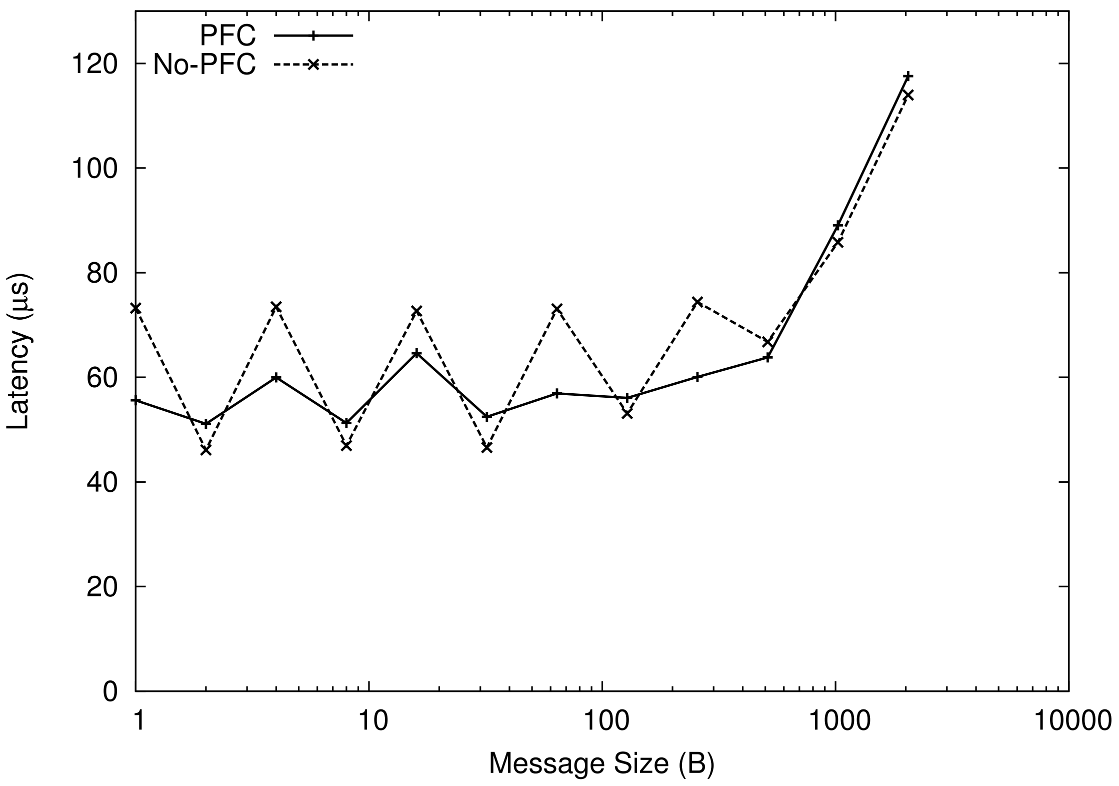


Figure 5-7: MPI exchange benchmark latency (short messages only)

size between the compute nodes from 0 to 4,194,304 bytes. There are many other benchmarks that are included with Intel MPI Benchmarks. We select the Exchange benchmark simply because it is the most demonstrative of the behavior that all of the benchmarks appear to have.

Figure 5-6 shows the entire results of the benchmark, while Figure 5-7 shows the results of the same benchmark but only uses the results from message sizes below 4,096 bytes so we can observe the small message size results better. Figure 5-6 shows that after message sizes of 131,072 bytes the latency of the PFC enabled network is consistent and increases with message size, while the latency of the network with PFC disabled jumps up dramatically and is very inconsistent. Figure 5-7 shows the PFC enabled network continues to have consistent and stable latency results across message sizes, while the PFC disabled network is inconsistent and fluctuates from message size to message size.

The results of the MPI experiment shows that MPI can benefit dramatically from using PFC on a network. Both the low end and full results clearly show that latency is significantly more consistent and stable across all message sizes. The results show that without PFC enabled on the network, MPI relies solely on TCP windowing and retransmission for flow control. Many in the industry believe TCP is a well established protocol that performs adequately, but we are able to show the significant benefits that can be had over TCP. We observe that when TCP retransmission occurs, large latency penalties are incurred. With PFC enabled on the network, retransmission is avoided and the latency is much more stable.

5.7 Throughput of iSCSI on a Converged Network

During the Throughput of iSCSI on a Converged Network experiments, only the CNA 1 and CNA 3 devices are used. In this experiment, the open source implemen-

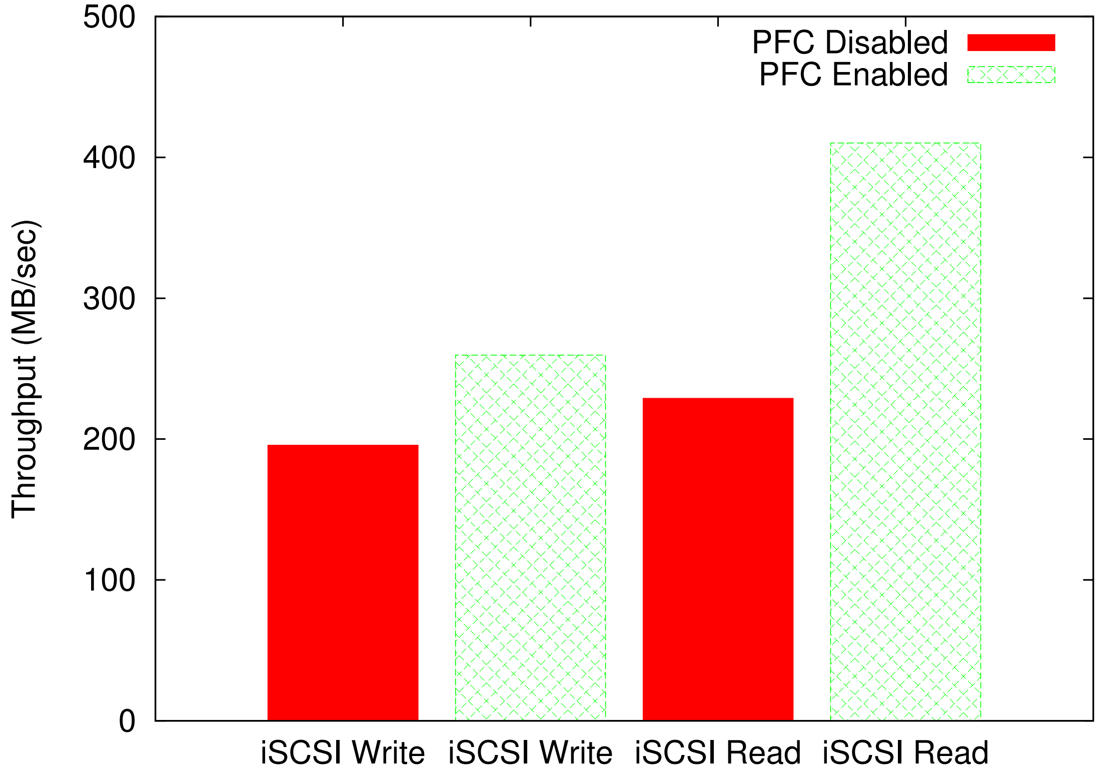


Figure 5-8: Throughput of different iSCSI operations with and without PFC enabled

tations of UNH iSCSI target and initiator are used [70]. Memory mode is used so the results will focus on the network and not be limited by hard drive throughput. CNA 1 is the iSCSI initiator and CNA 3 is the iSCSI target and throughput is measured from the initiator. Many in the industry think iSCSI will not benefit from DCB due to already having a mature flow control mechanism in TCP. For this reason, we wanted to look at iSCSI in the simplest possible configuration and see if there were any benefits by turning on PFC.

Figure 5-8 shows the results of iSCSI read and write operations. First the throughput is measured with PFC disabled and then with PFC enabled. The results show that, by enabling PFC on the network, throughput is dramatically increased. In the case of iSCSI reads, throughput is almost doubled. In the case of iSCSI writes, throughput is increased by over 25%. This can be explained with an understanding of how TCP flow control works. TCP flow control works by transmitting data until a packet is dropped at which point a back off occurs. After the back off, TCP will then slowly ramp up its throughput again. The use of PFC can significantly increase utilization of the available bandwidth on the network by avoiding the back off and the slow ramp up.

The results of the iSCSI experiment confirm the results from the MPI experiment. The throughput of both iSCSI read and write operations were significantly improved. By avoiding the overhead associated with TCP flow control, including TCP windowing and frame retransmission, throughput is increased by 25% in the case of write operations and up to 100% in the case of read operations.

5.8 Conclusion

The popularity of Fibre Channel over Ethernet and Data Center Bridging technologies is growing rapidly, so performance analysis of the base technologies within DCB is very important. Using devices from Intel and switches that have implemented some of these technologies, this chapter analyzes the performance characteristics of a data center grade network with common links as would be seen when using the Spanning Tree Protocol. The goal is to understand the characteristics of PFC. Without PFC, the system drives over 800 MB/s of throughput and loses close to 40% of the data. With PFC turned on, the throughput is reduced significantly to about 500 MB/s and frame loss no longer occurs. PFC works as expected in both a simple single traffic source to traffic sink scenario and two traffic streams scenario. When the throughput of one traffic stream is temporarily reduced however, it greatly reduces the throughput of other traffic streams in the same priority. When a high rate traffic stream was converged with a low throughput stream the low throughput stream was reduced as well. Surprisingly, when the high throughput stream is temporarily reduced it increases the throughput of the low throughput stream even though more PFCs were seen on the inter-switch link.

MPI is one of the most common applications used in computing cluster research. Typically, MPI relies on small messages to communicate between the compute node processors. Thus, latency is generally more important to MPI performance than throughput. Without PFC enabled, the MPI system relies on TCP windowing and retransmission for flow control which created very inconsistent latency results over the range of message sizes. When PFC was enabled, the overall latency was reduced and became very stable across all message sizes. Surprisingly, we were able to show that DCB can greatly benefit the latency of an MPI system even though TCP is a well established protocol that some in the industry believe would not benefit from DCB.

Computing clusters often utilize iSCSI or Fibre Channel to provide large amounts of fast storage for research. iSCSI is the Ethernet storage application of choice while

FCoE begins to become available. iSCSI relies on TCP flow control similar to MPI. Our iSCSI results confirm the results from the MPI experiment. When PFC is enabled so that TCP flow control and overhead is avoided, significant throughput improvements can be seen.

Future work needs to look at incorporating ETS, multiple priorities and CN functions. Experiments utilizing larger networks with multiple traffic streams on different priorities with ETS was not considered here due to a desire to examine how a common link in a Spanning Tree network performed with PFC. The lack of CN support limited the extent of our study here, but new implementations should support CN this coming year. The IEEE 802.1Qau standard has been completed and is moving to final publication at the time of this writing, so analysis of upcoming implementations will be very important.

Chapter 6

Deficit Round Robin Scheduling With Adaptive Weight Control

The previous chapter examined how DCB enabled networks can affect the performance of some common data center applications. While the previous chapter looked at overall application classes, it is important to start to explore the performance of multiple traffic streams within application classes. Many current data center switches are now implementing Deficit Round Robin in order to schedule traffic streams within traffic classes. This chapter examines the performance impact of DRR on current data center switches and identifies a problem. After exploring the fundamental mechanisms behind the problem, we propose a new DRR algorithm, implement it in hardware, and measure the impact the algorithm has on fairness within the network.

6.1 Introduction

Data center grade switches are now implementing deficit round robin (DRR) scheduling over the traditional weighted fair queuing (WFQ). DRR is simple to implement and has shown a close approximation to WFQ in terms of fairness. This chapter identifies a fairness issue using experiments on real data center hardware and proposes a new DRR scheduling algorithm that actively monitors the incoming traffic streams and adjusts weights accordingly to maintain fairness. We implement the Deficit Round Robin With Active Weight Control (DRR-AWC) scheduling algorithm in a commercial data center switch and fairness is observed while running the new algorithm. It is shown that DRR-AWC provides a significant fairness increase for traffic flows with small frames.

Scheduling algorithms and queuing theory are a balancing act of fairness, performance and simplicity of implementation. For the last twenty years, research has focused on Weighted Fair Queuing (WFQ), a well accepted scheduling algorithm, that offers both high performance and is optimally fair [46]. Unfortunately, WFQ has been shown to be overly complex and expensive to implement. Due to this complexity, Deficit Round Robin (DRR) was proposed shortly after WFQ was introduced [47].

Since the introduction of DRR many different researchers have examined the scheduling algorithm and proposed modifications to it. Some of the variations proposed are: Custom Deficit Round Robin [71], List-based Weighted Round Robin [72], Multiclass Round Robin [72], Dynamic Deficit Round Robin [73] and Deficit Round Robin with Fragmentation [74]. One of the more popular variations of DRR is Weighted DRR which allows the ability to set a different Quantum Value per input queue.

Currently many switches now implement Weighted DRR since it is easier to implement and provides a close approximation to the fairness of WFQ. Specifically, many of the data center switch implementations have begun to support Weighted DRR as their scheduling algorithm of choice.

Today's networks are an ever evolving ecosystem of different traffic classes. As recently as only ten years ago web traffic, email and Voice over IP (VoIP) traffic dominated traffic patterns [75]. The creation of cloud computing has ushered in a new wave of computing requirements in datacenters. Modern datacenters are focused on shared processing power as well as storage traffic. The Message Passing Interface (MPI) is the predominant protocol for sharing processing power and is made up of mostly small frames (less than 256 bytes). Storage and video traffic streams are characterized by large packets and very long transfers. This dichotomy has been described in terms of length as dragonflies (short bursts) vs. tortoises (long and sustained) and in terms of size as mice vs. elephants [76].

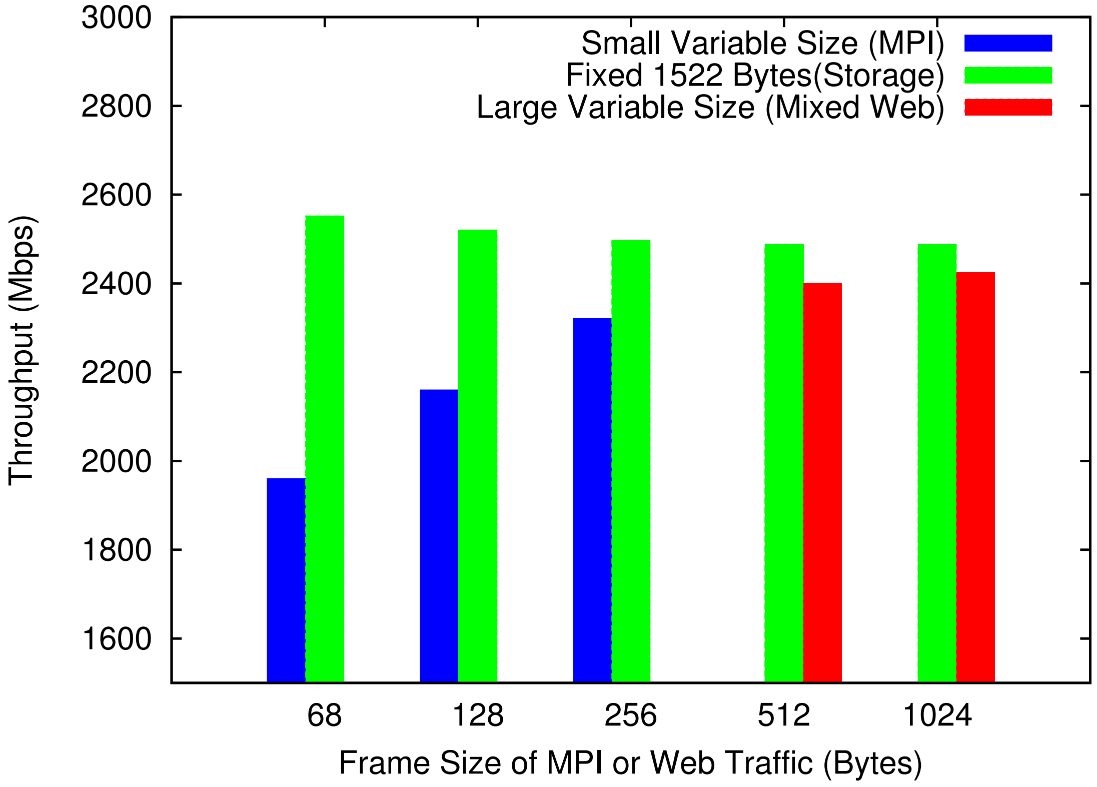


Figure 6-1: Throughput of typical traffic classes

With such a diversity of traffic classes it is easy to see the importance of studying existing scheduling algorithms within modern day data centers. It is important to ensure that traffic streams of small frames as well as large frames each utilize the network fairly. Figure 6-1 shows the actual throughput of different frame sizes converged in a data center network. Along the x-axis you can see how small variable frames and medium Mixed Web Traffic frames match up against large storage frames that always have a max size of 1522 bytes. In the most extreme case, 68 bytes, the small MPI frames are receiving almost 25% less throughput than the storage traffic class.

On a 10 Gbps network there is a minimum of 12-bytes of inter-frame gap and 8-bytes of preamble that are required to be inserted between each transmitted frame in order to allow a receiver to properly receive the frames. Small frames already have a usable data penalty due to overhead associated with headers, so by counting inter-frame gap and preamble in the scheduling algorithm as well, small frame streams are receiving an additional 29% penalty in throughput as well. With normal frame overhead as well as 10 Gbps line overhead, small frames will see well over 30% less usable data throughput than larger frames.

We will provide results from further experimentation that shows exactly how the unfairness manifests. Following that, a new variation of the DRR algorithm that resolves the issue is proposed. Then the new algorithm is implemented in a switch and results of the algorithm working directly are presented. We will measure the throughput of traffic streams while varying the frame sizes of traffic streams and observe that the algorithm causes the throughput of the streams to converge.

This chapter will provide an immediate solution for the many switches that currently implement DRR scheduling algorithms and expand the research understanding of how traditional scheduling algorithms work in a modern data center.

The motivation of this chapter is two fold:

1. To understand the interaction between an existing scheduling algorithm and data center technologies.

2. To propose a new deficit round robin based scheduling algorithm that resolves fairness issues with deficit round robin scheduling on data center hardware.

The rest of this chapter is organized as follows: Section 6.2 presents the experimental results and analysis. Section 6.3 presents the proposed algorithm along with a discussion of implementation choices. Section 6.4 presents an evaluation of the performance and fairness of the implemented algorithm. Section 6.5 summarizes our conclusions about the current study and outlines ideas for future research.

6.2 Experimental Results

The objective of the following series of experiments is to evaluate the performance of DRR on data center grade equipment. The motivation for this work is to understand exactly how the unfairness of small frames on a network with DRR enabled manifests. Initial experiments have shown significant throughput differences between traffic streams of different frame sizes. Therefore, we developed a method to adjust the weights of different queues in the switch and measure the throughput using test hardware from Ixia. Through an iterative process, we perform several experiments to adjust the weight of the queues until the throughput of each stream was equivalent.

The experimental setup is shown in Figure 6-2. The switch is an early release 10 Gbps cut-through programmable switch that provides near wire speed transmission on the network. The topology shown in Figure 6-2 was chosen because it is the least complex topology that provides a congested link and will not obfuscate any of our results with additional network complexities.

Traffic was generated with the Ixia XM2 Chassis with LSM10GXM4XP load modules. Utilizing the Ixia test ports allowed us to quickly and easily configure different traffic streams. The Ixia ports are able to generate line rate 10 Gbps traffic at even small frame sizes. With this setup we are able to focus our experimentation on the

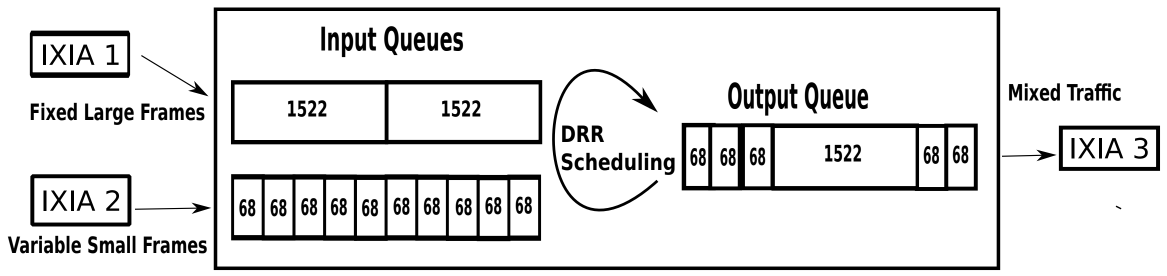


Figure 6-2: Experimental setup

switch and the scheduling algorithm without concern for other load generation issues. All traffic streams in the following experiments use Layer 2 traffic in order to avoid congestion mechanisms of higher layers and allow us to focus on the scheduling algorithms of the switch.

6.2.1 Experiment 1: jumbo frames

The first experiment utilizes all three of the Ixia ports in the test setup. A large frame traffic stream (Traffic Stream 1) is generated from Ixia Port 1 to Ixia Port 3 that consists of either 1522, 2500 or 5000 byte frames. Each of the three graphs (Figures 6-3, 6-4, 6-5) corresponds to a different frame size for Traffic Stream 1. The second stream (Traffic Stream 2) is generated from Ixia Port 2 to Ixia Port 3 that consists of variable sized small frames. This setup results in both streams needing to be scheduled via DRR on the switch port connected to Ixia Port 3.

The goal of this experiment is to find the scheduling weight of Traffic Stream 2 required to make the throughput of each traffic stream equal. The ratio of the weight for Traffic Stream 2 to the weight for Traffic Stream 1 is plotted on the y-axis and the frame size of Traffic Stream 2 is plotted on the x-axis. The results show both the range at each data point and the weight required to create a fair throughput situation. The results of Figure 6-3, Figure 6-4 and Figure 6-5 show smaller frames are penalized

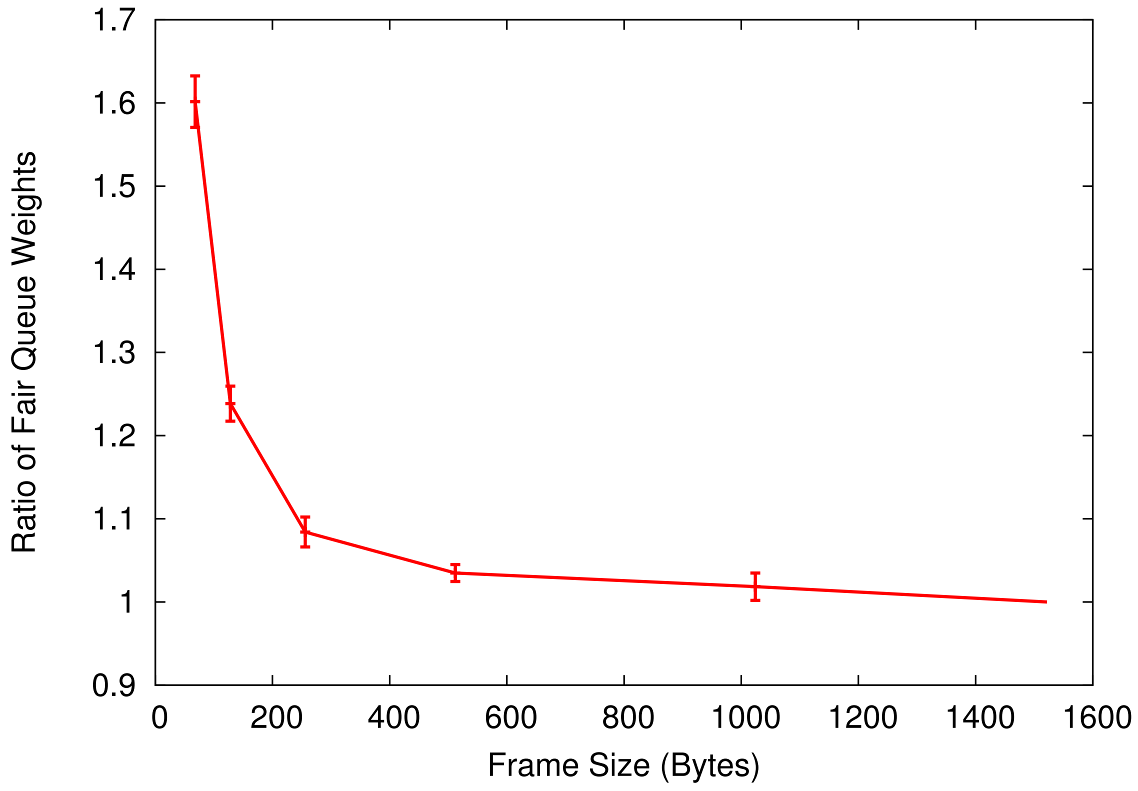


Figure 6-3: Experiment 1 - ratio of traffic stream 2 fair queue weight to traffic stream 1 fair queue weight with 1522-byte frames

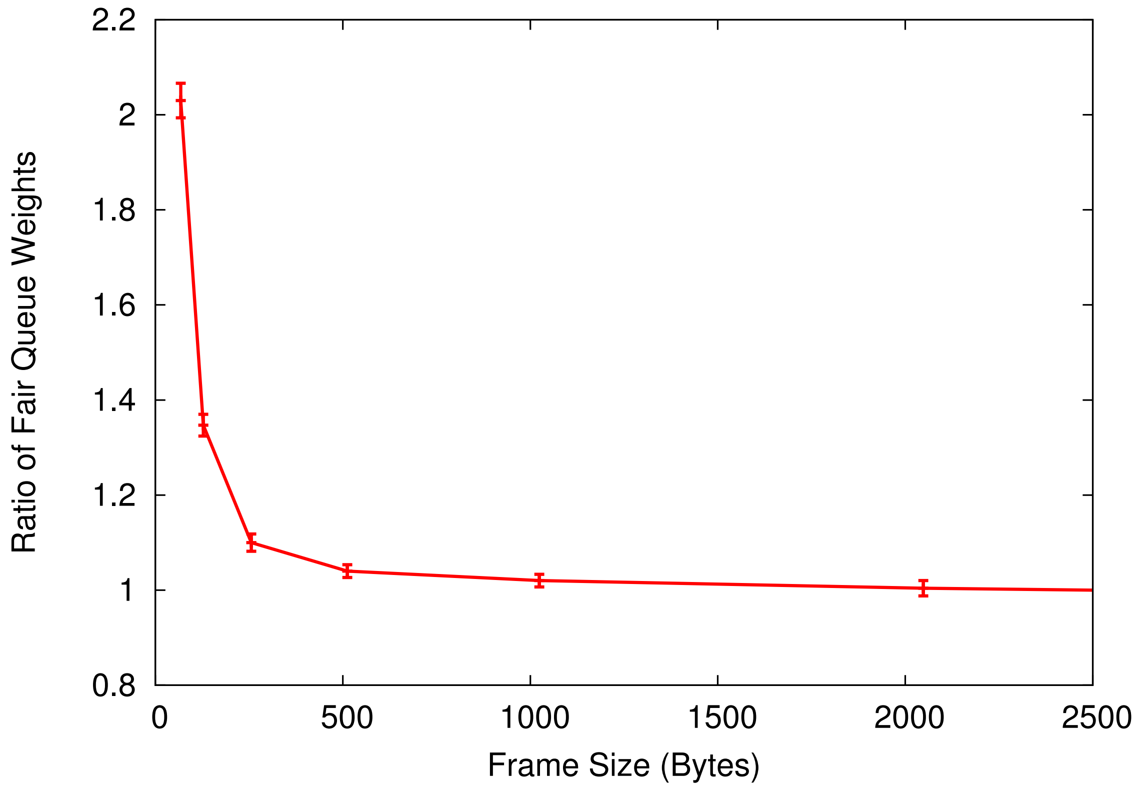


Figure 6-4: Experiment 1 - ratio of traffic stream 2 fair queue weight to traffic stream 1 fair queue weight with 2500-byte frames

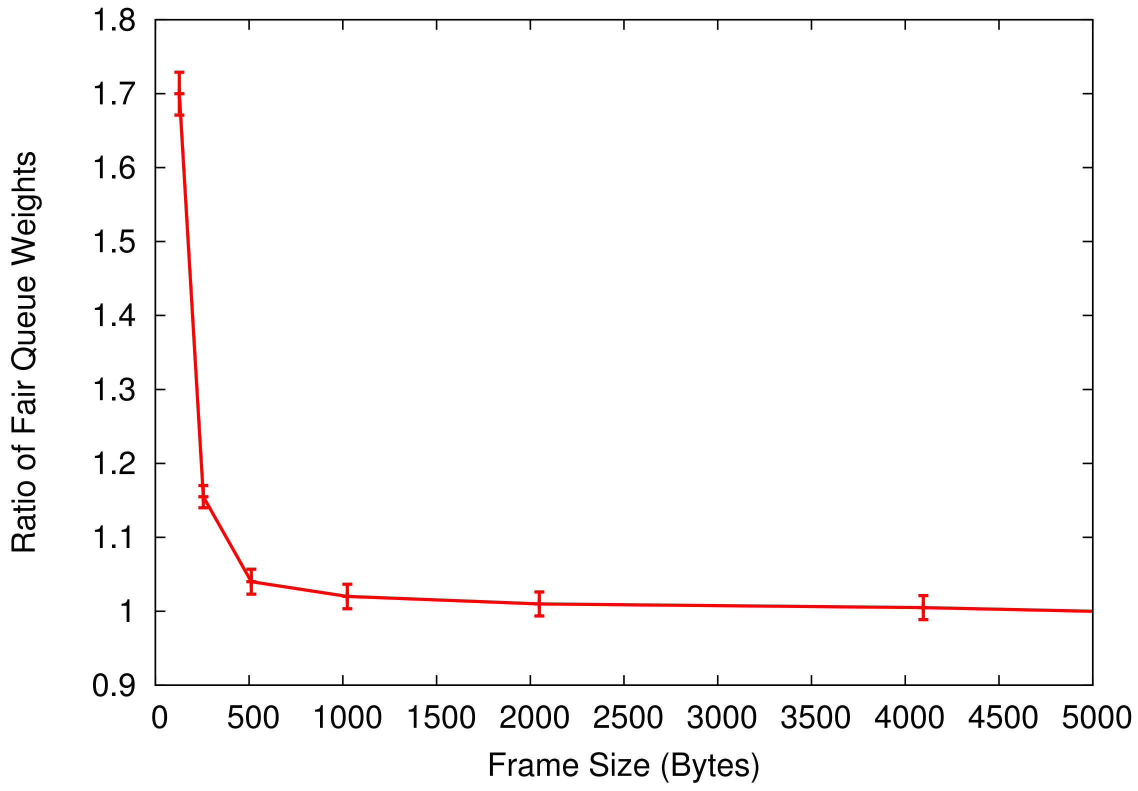


Figure 6-5: Experiment 1 - ratio of traffic stream 2 fair queue weight to traffic stream 1 fair queue weight with 5000-byte frames

greatly and require significantly higher weights in order to equal the throughput of Traffic Stream 1. Once Traffic Stream 2 reaches a frame size between 200 and 500 bytes, the penalty begins to tail off and slowly approaches the weight of Traffic Stream 1. For the smallest frames, the weight required to create fair throughput is almost double the weight of Traffic Stream 1.

6.2.2 Experiment 2: multiple input queues

In the second set of experiments, we investigate when multiple small frame input queues were used so we connected a fourth Ixia port to the switch. Instead of using only one input queue of small 68-byte frames and one input queue of large frames, we use two input queues of small frames of different sizes (68, 128, 256, 512, 1024 bytes) along with the input queue of large frames. We plot the results from Figure 6-3 in the graph to allow comparison with the new results. The height of the bars indicates the scheduling weight of each input queue required to reach fair throughput with the large frame stream and the location of the bar on the x-axis indicates the frame size of the input queue. The solid line is the same line from Figure 6-3 included here as a reference point, to show that the new results completely agree with the results from the previous experiment.

Figure 6-6 and Figure 6-7 show that the results of Experiment 2 completely coincide with the results of Experiment 1. Whether there is a single queue or multiple input queues of small frames, the weight required to achieve the same throughput as the large frames traffic stream is the same.

6.2.3 Experiment 3: single input queue with mixed frame size

In the third experiment, we want to understand how mixed traffic within a queue affects the results. The input queue of large frames was maintained, but now for the single small frame queue we cause 50% of the frames to be of one size (x bytes) and

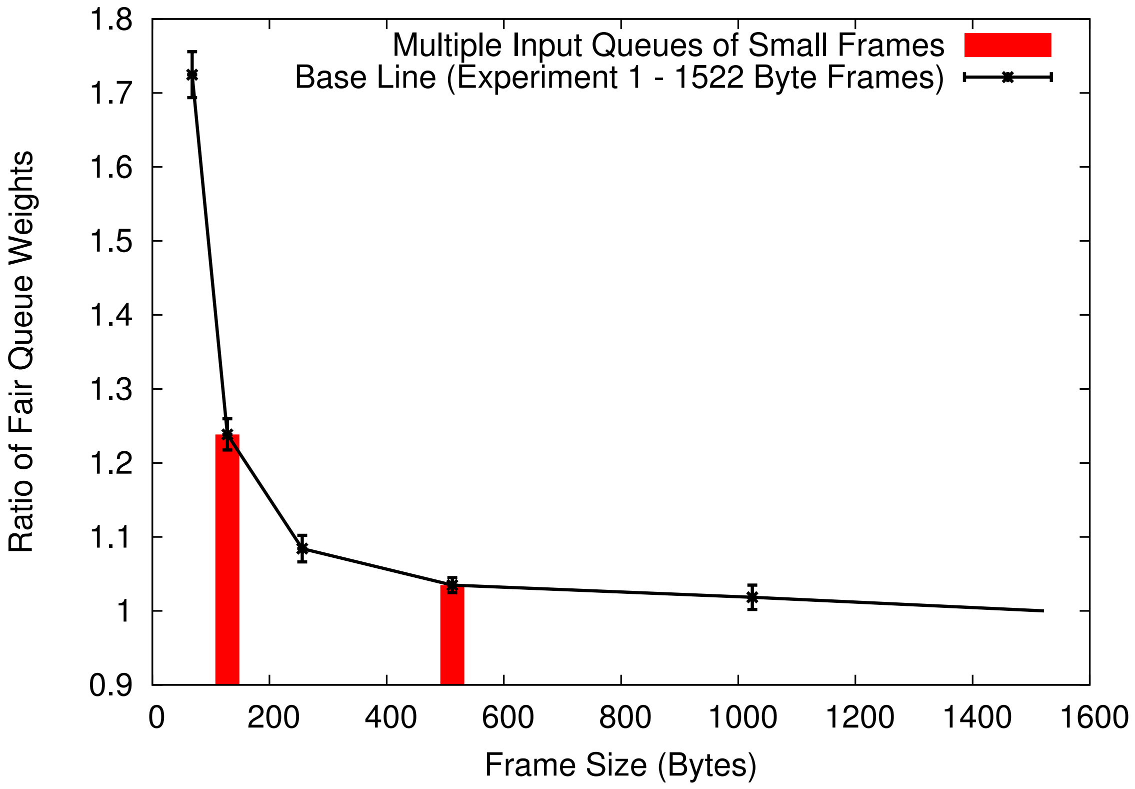


Figure 6-6: Experiment 2 - multiple input queues with different frame sizes

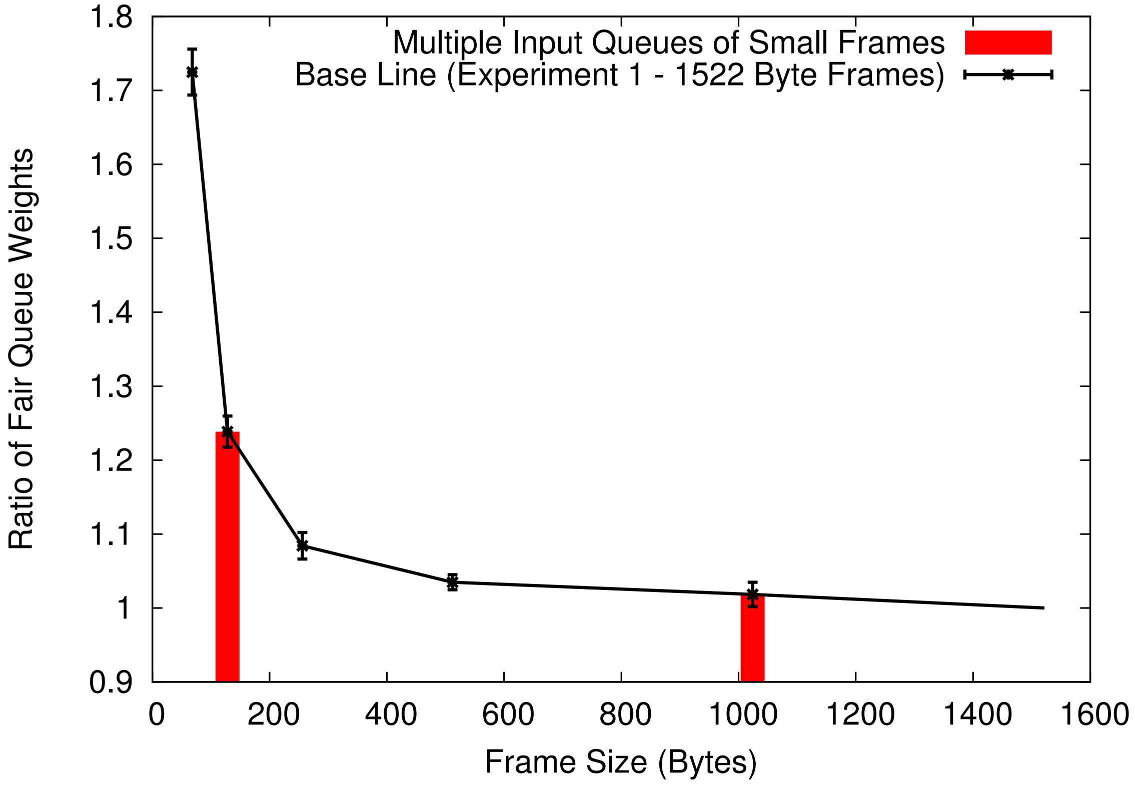


Figure 6-7: Experiment 2 - multiple input queues with different frame sizes

50% of the frames to be of another size (y bytes). We vary the sizes of each of the small frames and plot the results in Figure 6-8.

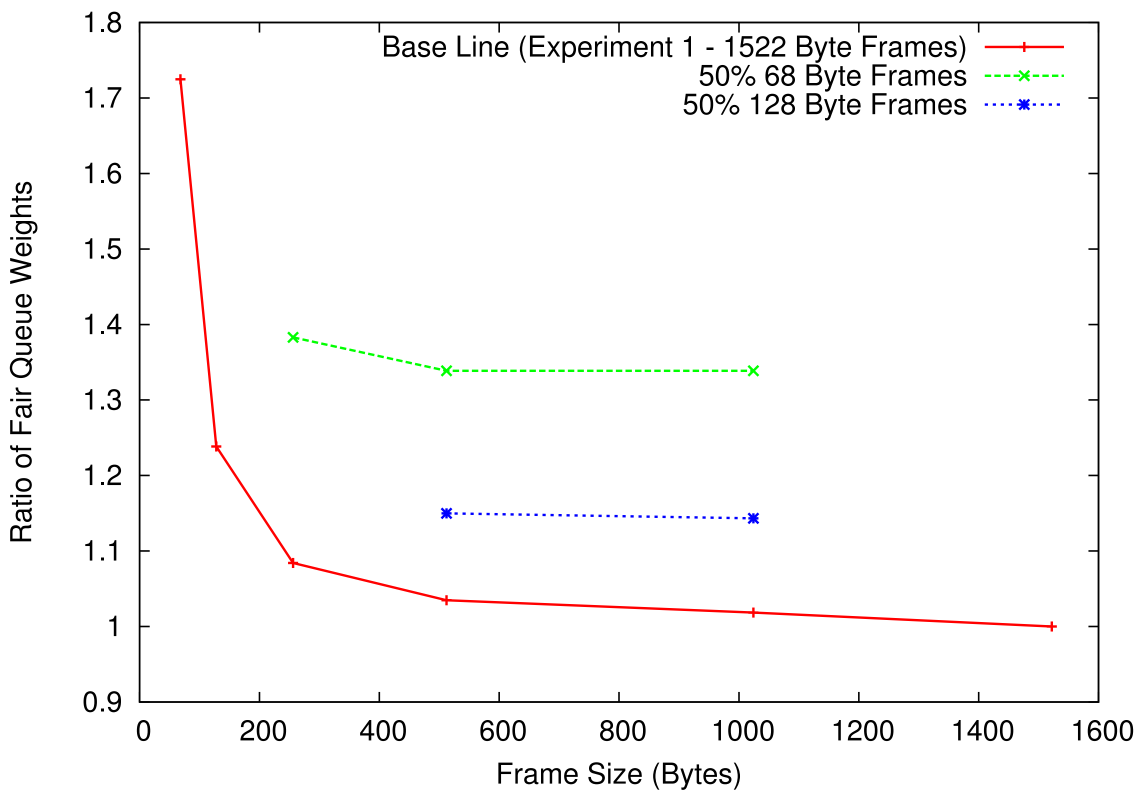


Figure 6-8: Experiment 3 - single input queue with mixed frame size traffic

From Figure 6-8 it can be seen that we perform two series of tests. In the first series, 50% of the small frames in the queue were 68-byte frames and the size of the other 50% of the queue varied from 256 to 1024 bytes. In the second series, 50% of the small frames were 128-byte frames and the other 50% varied from 512 to 1024 bytes. The third line, or Base Line, is again the result from Experiment 1 used as a reference point. Figure 6-8 shows that the mixed input queue traffic results in an averaging of the results from the previous experiments. For example, when 50% of the input queue was 68-byte frames and the other 50% of the queue was 256-byte frames it is discovered that the ultimate result (1.39) is about the average (1.4) of the result found in Experiment 1 for 68-byte frames (1.72) and 256-byte frames (1.08).

6.2.4 Trace analysis

After initial experimentation to understand the extent of the unfairness issue, we moved to detailed analysis of traces. A Wireshark^{TM} capture was taken of each port before and after adjusting the weights using an Absolute Analysis Investigator Protocol Analyzer. Each trace was analyzed to understand the pattern of frame transmission. It was clear from the traces that significantly fewer small frames were being transmitted for every large frame in the system. Section 6.3 proposes a solution to this problem.

6.3 Deficit Round Robin With Active Weight Control Algorithm

The results from Section 6.2 provides enough information to determine the optimal weight for a given frame size. The objective of this section is to propose a simple and accurate method to calculate appropriate weights for different traffic flows. The proposed algorithm relies on a switch's statistic counters, which are fairly common and keep track of the number of frames that have been seen of different sizes and groups a range of sizes into bins (stat bins). Generally speaking, the algorithm goes through the switch's stat bins for each input queue in the switch and adjusts the DRR weight of the input queue based on the proportion of small frames. Input queues with smaller frames will get their weights adjusted higher than input queues with large frames. This will cause the throughput of the different input queues to become close to equal thus ensuring fairness across all of the input queues regardless of the average frame size in each input queue.

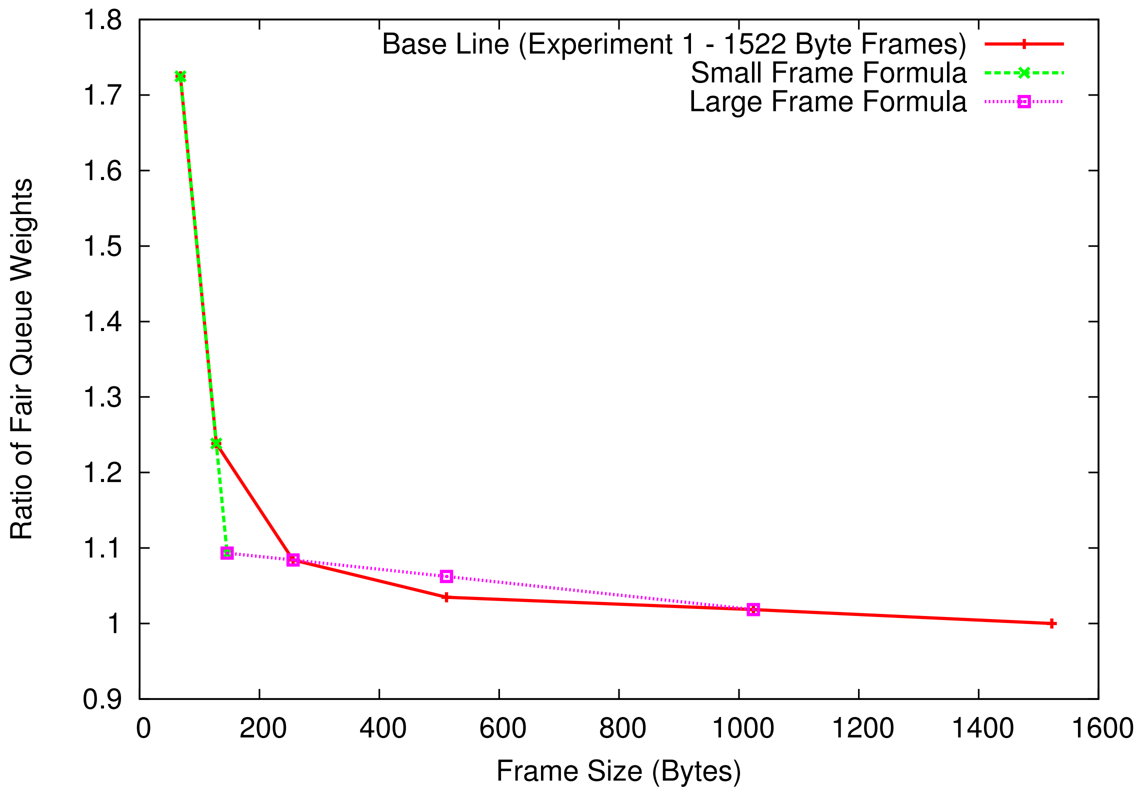


Figure 6-9: Small and large frame formulae

Formula 6.1 shows the formula for traffic streams with frames smaller than 146 bytes where x is the average frame size and $f(x)$ is the scheduling weight. Formula 6.2 shows the formula for traffic streams with frames larger than 146 bytes. 146 bytes is the exact pivot point calculated from the previous results that provides the least amount of deviation from the original curve and maximizes fairness using line fitting. Figure 6-9 shows both the Small and Large Frame Formulae in relation to the base line. The formulae are derived by using line fitting from the baseline results. Algorithm 1 shows the pseudocode of the DRR-AWC algorithm that we actually implemented on the switch. The switch implementation algorithm is limited by the capabilities that the switch provides to programmers. One of the major limitations is that it only provides frame size categories (or bins) so the ultimate accuracy is limited by the granularity of each of the bins. With more bins the algorithm could be finer grained and more accurate for frame sizes not explicitly identified by the bins. The switch utilized in the experiments currently supports five bins for frames between 68 bytes and 1024 bytes (68, 69-128, 129-256, 257-512, 513-1024 bytes). The number of frames in each bin are reported for each input queue on the switch.

$$f_s(x) = -0.008x + 2.276 \tag{6.1}$$

$$f_l(x) = -8.555 \cdot 10^{-5}x + 1.106 \tag{6.2}$$

The algorithm begins with clearing the traffic statistics, accumulating traffic statistics for one minute, then getting the statistics again. This provides a running average of the number of frames for each frame's size category per minute. The one minute to accumulate traffic statistics was chosen as a balance between consuming too much of the switch's processing resources and providing adequate fairness over time. The algorithm then goes through each bin and adds the proportion of the weight to the total weight based on the number of frames in each bin and the fair weight value provided by Formula 6.1 and Formula 6.2. If the frame size is 146 or less the proportion of weight added to *totalWeight* is determined using the small frame algorithm (Formula 6.1), otherwise it uses the large frame algorithm (Formula 6.2). Finally, the total weight is multiplied by the maximum frame size and the weight of the input queue is set for each input queue in the switch. During the next scheduling round each input queue is then scheduled based on the new weight calculated.

Algorithm 1 Switch implementation

```
while true do
    clearStats()
    accumulateTrafficStats(1 minute)
    list statsBin = getStats()
    int totalWeight = 0
    for each frameCountStat in statsBin do
        if frameSize <= 146 then
            totalWeight += $f_s$(frameSize)$\cdot \frac{\text{numberOfFramesPerSizeBin}}{\text{totalFramesOfAllBins}}$
        else
            totalWeight += $f_l$(frameSize)$\cdot \frac{\text{numberOfFramesPerSizeBin}}{\text{totalFramesOfAllBins}}$
        end if
    end for
    setWeight(totalWeight $\cdot$ maxFrameSize)
end while
```

6.4 Evaluation

After implementing the proposed DRR-AWC algorithm in the commercial data center switch and running some additional tests, we looked at how the algorithm has improved the fairness of the system. We re-ran the previous experiments once with the algorithm turned off (using only the switch's DRR) and a second time with the proposed DRR-AWC algorithm turned on and compared the results. In order to understand the improvements to fairness, we used Jain's Fairness Index (Formula 6.3). Jain's Fairness Index provides a measure of fairness ranging from $1/n$, where n is the number of "users", to one with one being the most fair [77]:

$$J = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \tag{6.3}$$

where x is the throughput and n is the number of input queues.

Figure 6-10 show the results of this analysis. The bars labeled DRR are the calculation of Jain's Fairness Index when the switch is running the base DRR scheduling algorithm. The bars labeled DRR-AWC are the fairness calculation when the switch is running the proposed scheduling algorithm. It can be seen that the fairness increases significantly, in some cases from less than 0.985 to over 0.999. Jain's Fairness Index has an emphasis on starvation, so the values are not largely different over the range of the index but from 0.985 to 0.999 represents a real world throughput improvement of almost 33%.

6.5 Conclusion

Cloud computing has created a new wave of shared computing within datacenters. MPI and storage traffic make up a large part of the traffic mix within modern datacenters. This creates a mix of traffic within the datacenter with both large and small

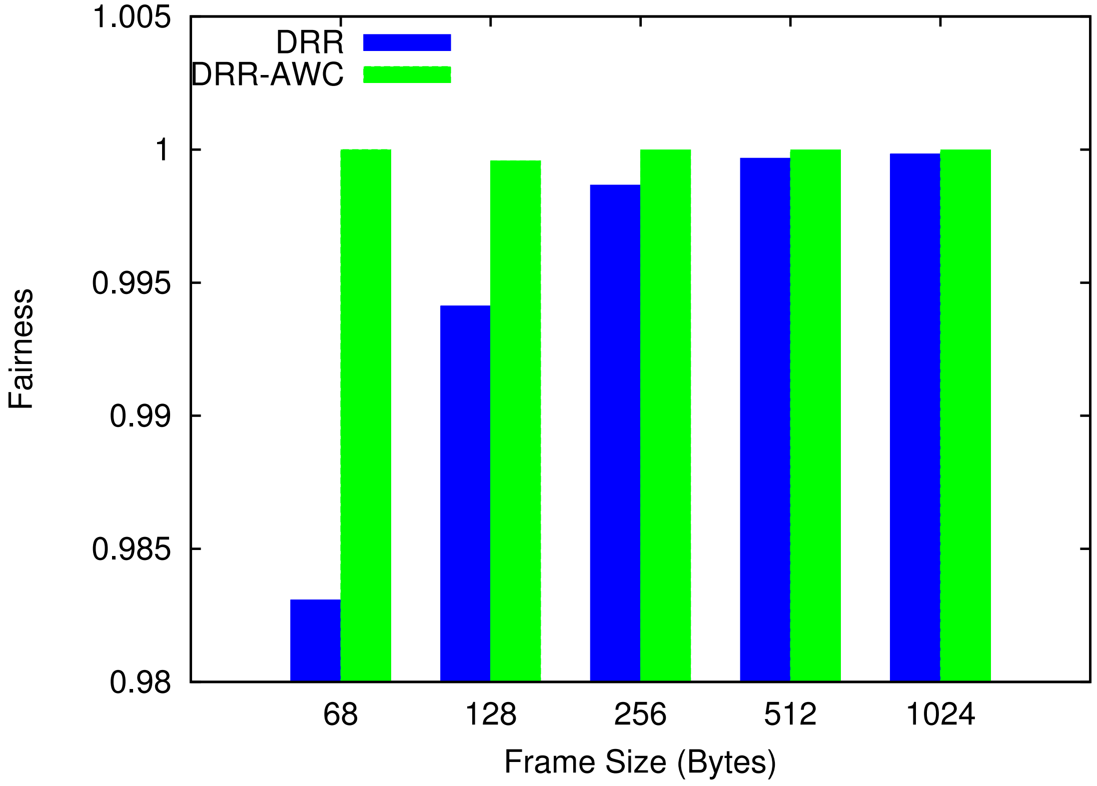


Figure 6-10: Fairness comparison of DRR and DRR-AWC

frames that will need to be fairly scheduled. The problem identified in this chapter is that smaller frames get a disproportionately smaller amount of the network throughput on datacenter grade 10 Gigabit Ethernet hardware.

In analyzing the discrepancy between the throughput of large frames and small frames, this study proposes a new DRR algorithm called Deficit Round Robin with Active Weight Control (DRR-AWC). The new algorithm involves monitoring the queues and frame sizes and periodically adjusting the weight of the DRR scheduling algorithm to compensate for the unfairness. We then implemented the algorithm in a commercial data center switch and evaluated the fairness. The new algorithm does not add greatly to the complexity of the existing scheduling algorithm, yet actively monitors traffic streams and adjusts queue weights in order to maximize fairness. We were able to show an increase in fairness as measured by the Jain's Fairness Index.

Chapter 7

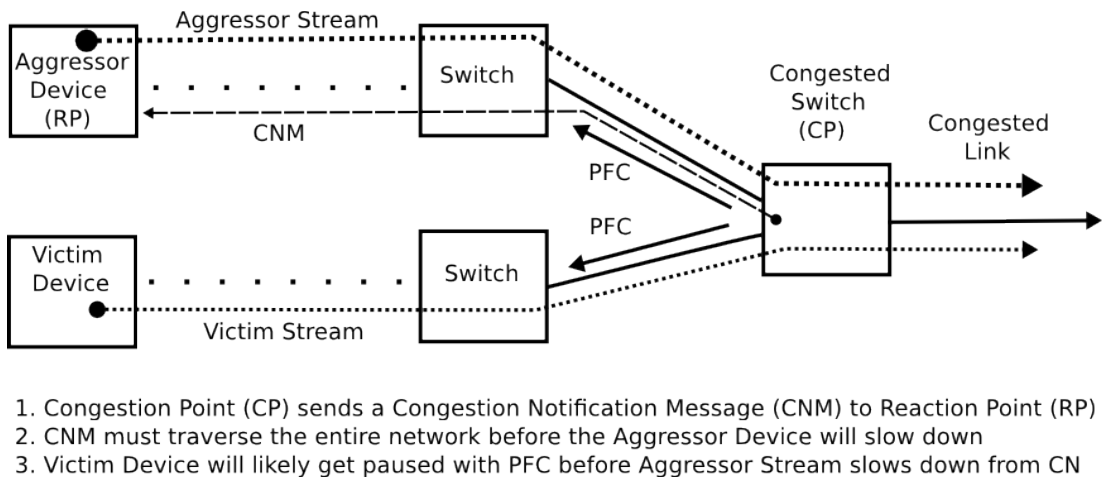
Targeted Priority-based Flow Control

The previous chapter examined the fairness of traditional scheduling algorithms interacting with new data center hardware. In this chapter, we continue to address fairness in the data center with a focus on the DCB protocol mechanisms that target aggressor streams within a traffic class, in order to prevent other traffic streams in that class from being caught in congestion when PFC is used on the network. Congestion notification (CN) is the mechanism defined to target aggressor streams in DCB. It is a well defined solution that is complex to implement and suffers from network round trip time feedback latency. We develop and propose a new targeting mechanism called Targeted priority-based flow control (TPFC) that maintains the simplicity of PFC, while providing fast hop-by-hop response times.

7.1 Introduction

The IEEE Data Center Bridging (DCB) working group recently approved the standards for Priority-based Flow Control [1, 2] and Congestion Notification [20]. PFC allows the independent pausing of up to eight different traffic classes and is designed to prevent frame loss due to congestion. It operates on a link by link basis to respond quickly when congestion occurs. Earlier research has shown the benefits [78] and limitations [79] of PFC. Recent work has proposed a dynamic pause time calculation method and provided analysis of the benefits of adjusting the pause time [80]; however, in that paper all traffic is subject to the same dynamic pause time and, therefore, it does not directly offer a way to target aggressor streams.

Congestion Notification is an end-to-end congestion mechanism that attempts to tell the source of traffic to slow down before PFC forces the transmission to stop. CN by itself cannot guarantee frames are not lost. Early work showed that up to 10% of frames could be lost under moderate traffic [81]. An example of the feedback loop latency can be seen in Figure 7-1. The CN mechanism for targeting streams adopted by the IEEE is based on a scheme called Quantized Congestion Notification (QCN) that randomly samples the congested buffer and sends congestion messages to sources of traffic with greater frequency as the queue fills [82].



1. Congestion Point (CP) sends a Congestion Notification Message (CNM) to Reaction Point (RP)
2. CNM must traverse the entire network before the Aggressor Device will slow down
3. Victim Device will likely get paused with PFC before Aggressor Stream slows down from CN

Figure 7-1: CN feedback loop

7.2 Targeted Priority-based Flow Control

As discussed above, PFC grants the capability to pause independent traffic classes. Within a traffic class, paused streams suffer from *fate sharing*. Consider two types of streams: those that utilize less than their fair share of bandwidth and those that attempt to utilize more than their fair share of bandwidth. All streams in this traffic class will be paused by a PFC frame, even though some are utilizing less than their fair share. These *victim streams* suffer the same fate as those streams utilizing more than their fair share, so called *aggressor streams*.

Congestion Notification (CN) was proposed to address the fate sharing problem of PFC by ultimately reducing or eliminating the use of PFC. This chapter proposes a new mechanism based on the concepts within CN and PFC to minimize fate sharing. The proposed mechanism does so by detecting congestion early and sending PFC frames only to link partners transmitting aggressor streams, while maintaining compatibility with CN. By transmitting PFC frames only to link partners transmitting aggressor streams, fate sharing is eliminated on any victim stream transmitted by any other link partner. The following discussion focuses first on how to detect congestion early, then on how aggressor streams are identified.

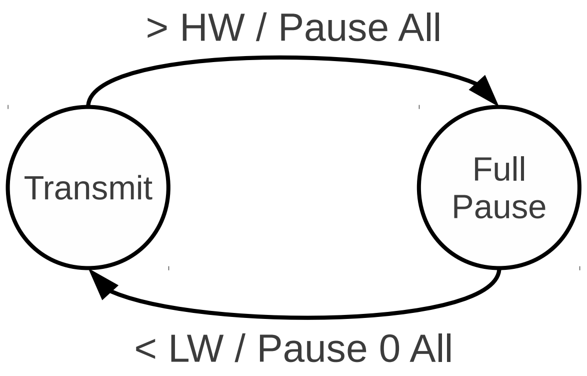


Figure 7-2: State diagram of PFC link states

In the current PFC mechanism, a queue implements a low watermark (LW) and a high watermark (HW). When the queue fills to the HW, a PFC frame is sent to all link partners. When the queue drains to the LW it sends a PFC frame with a pause time of zero in order to allow the link partners to transmit again (Figure 7-2). The LW must be set such that the queue does not completely drain so that the congested link does not become underutilized. The HW must be set such that the queue maintains enough margin to accommodate all possible frames in flight, preventing frame loss. This chapter proposes to add a third watermark called the *target watermark* (TW). When the queue fills to the TW, a PFC frame is sent only to link partners with aggressor traffic streams (Figure 7-4). Implementation of aggressor stream identification requires keeping track of the ingress port for each traffic stream, a requirement that is already present in CN. Figure 7-3 shows the queue layout with each of the watermarks defined. The TW is set so that there is enough queue space to handle frames in flight on the link:

$$TW = N \cdot F \tag{7.1}$$

where N is the number of link partners with aggressor streams to be targeted and F is the number of frames in flight for each link partner with aggressor streams. This allows implementors to choose how many link partners with aggressor streams they wish to target, N, as a trade off between cost and capability.

The next question is how to determine which link partners have aggressor streams. This chapter proposes two mechanisms: a random sampling approach [20] and a fair bandwidth approach. Similar to how QCN targets streams, the first proposed approach randomly samples a frame from the queue when the TW is reached and sends a PFC frame to the link partner that sent the sampled frame. By randomly sampling the queue, link partners with aggressor streams have a higher chance of being sampled and paused. In the fair bandwidth approach, the bandwidth assigned

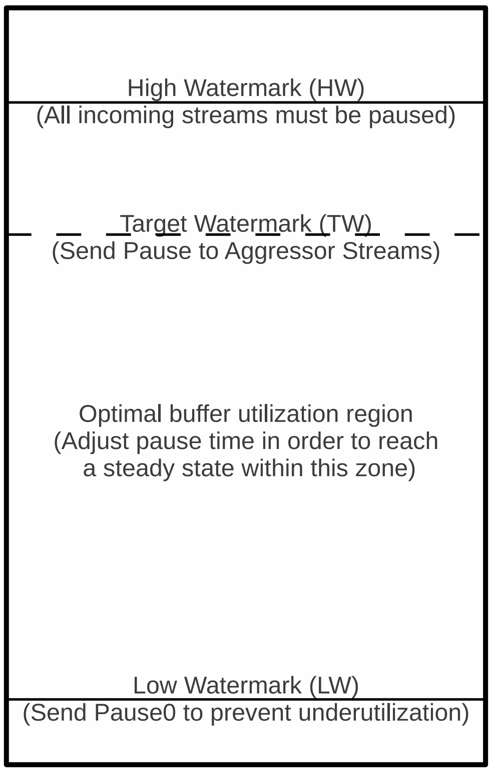

High Watermark (HW)
(All incoming streams must be paused)
Target Watermark (TW)
(Send Pause to Aggressor Streams)
Optimal buffer utilization region
(Adjust pause time in order to reach
a steady state within this zone)
Low Watermark (LW)
(Send Pause0 to prevent underutilization)

Figure 7-3: Targeted PFC Queue Layout

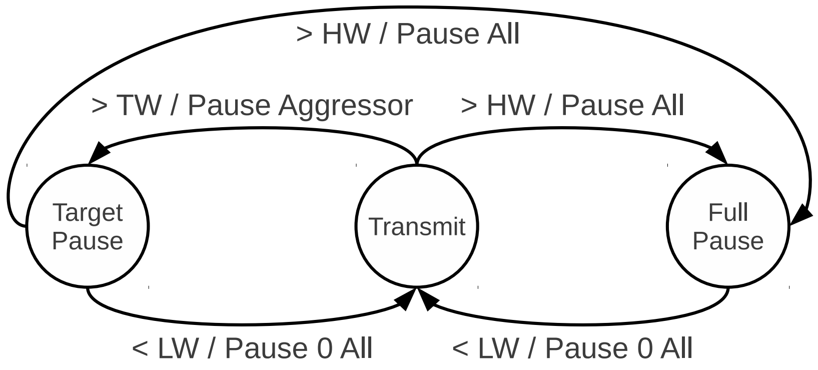


Figure 7-4: State diagram of TPFC link states

to stream i is calculated using the formula

$$b_i = \frac{b}{N} \tag{7.2}$$

where b_i is the bandwidth assigned to stream i, b is the total available bandwidth, and N is the number of different link partners that have sent frames to the queue. Any link partner exceeding this limit is deemed a link partner with aggressor streams and is paused when the TW is reached. The fair bandwidth mechanism is a simple fairness algorithm and future work can investigate more sophisticated ways to determine the link partners with aggressor streams. It is important to note that the proposed scheme is work-conserving such that if there are link partners with streams under-utilizing their fair share, the link partner with aggressor streams is allowed to over-utilize the fair bandwidth limit without being paused until the combined load exceeds the congested link's capacity, at which time the TW will be reached.

7.3 Experimental Setup

Utilizing the ns-2 simulator [83], a lossless queue that models the standard PFC behavior was implemented. After verifying the initial implementation, the behavior of PFC was extended with the proposed random sampling and fair bandwidth mechanisms. The topology used in the simulations is a simple congested link with three ingress ports and a single egress port. Since TPFC is a Layer 2 mechanism that operates in a hop-by-hop manner, more complex topologies do not provide any additional information. The traffic is limited to Layer 2 to avoid competing with other flow control mechanisms.

The memory model in the simulations follows the shared memory queue model. Traditional switches (see Figure 7-5) consist of an ingress queue, an egress queue and a switching element. This architecture worked well for a store-and-forward approach in

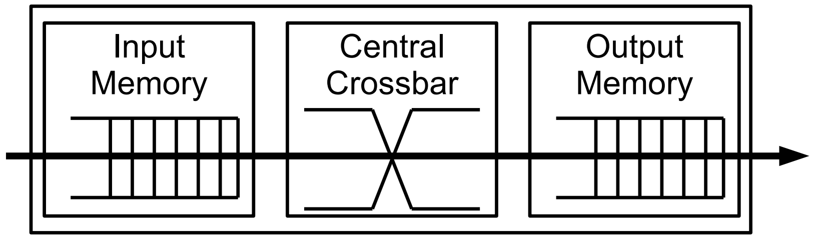


Figure 7-5: Traditional memory queue switch architecture

which packets are received in their entirety before being switched to their destination. A shared memory queue (see Figure 7-6) provides the benefits of easier memory management, fewer intra-switch packet copies and lower latencies. For all of these benefits, modern data center bridging switches are commonly shared memory queue architectures.

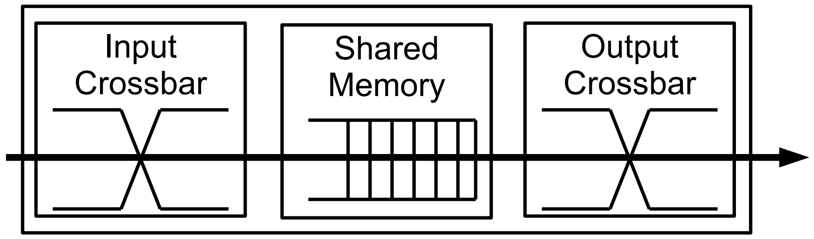


Figure 7-6: Shared memory queue switch architecture

The following parameters were used in the simulation.

Link Delay

Link delay was based on the formula used to determine how much buffer is needed above the HW in order to ensure no frame loss, and was taken from the IEEE standard [1]. This value represents a worst case delay between nodes. The total link delay is about 11 full size frames:

$$TotalDelay = 2M + P + 2C + 2I + H \tag{7.3}$$

where M is the maximum possible frame size, P is the size of a PFC frame, C is the cable delay, I is the delay caused by the interface, and H is the delay caused by higher layers. The formula assumes a port type of 10GBASE-SR, the predominant technology in the data center today, and assumes a 300 meter cable between nodes, the maximum length defined in the standard.

Queue Size

The size of each queue in the simulation was set to 100 frames.

Pause Time

The maximum pause time is about 3.4 ms as specified by the IEEE standard [2]:

$$MaximumPauseTime = 65,535Q \tag{7.4}$$

where Q is a pause quantum equal to 512 bit times at 10 Gbps line rate.

7.4 Results

The baseline PFC implementation was first validated by examining the congested output link to ensure that no frames were lost. Through the simulations, it was observed that by varying the HW frame loss could be eliminated. It was also determined that both the *Lossless-HW* and the *Lossless-HW-LW* implementations showed the desired lossless behavior.

The goal of the next experiment was to examine the throughput of the outbound link with three traffic sources starting simultaneous transmissions at 1 ms. Figure 7-7 shows the throughput of the outbound link. A lossy FIFO (without PFC) queue results in the full link utilization but is not lossless. Lossless-HW is lossless but results in the lowest link utilization of the outbound link. It can be seen that by implementing a LW (Lossless-HW-LW), the lossless queue is able to maintain close to full utilization of the link.

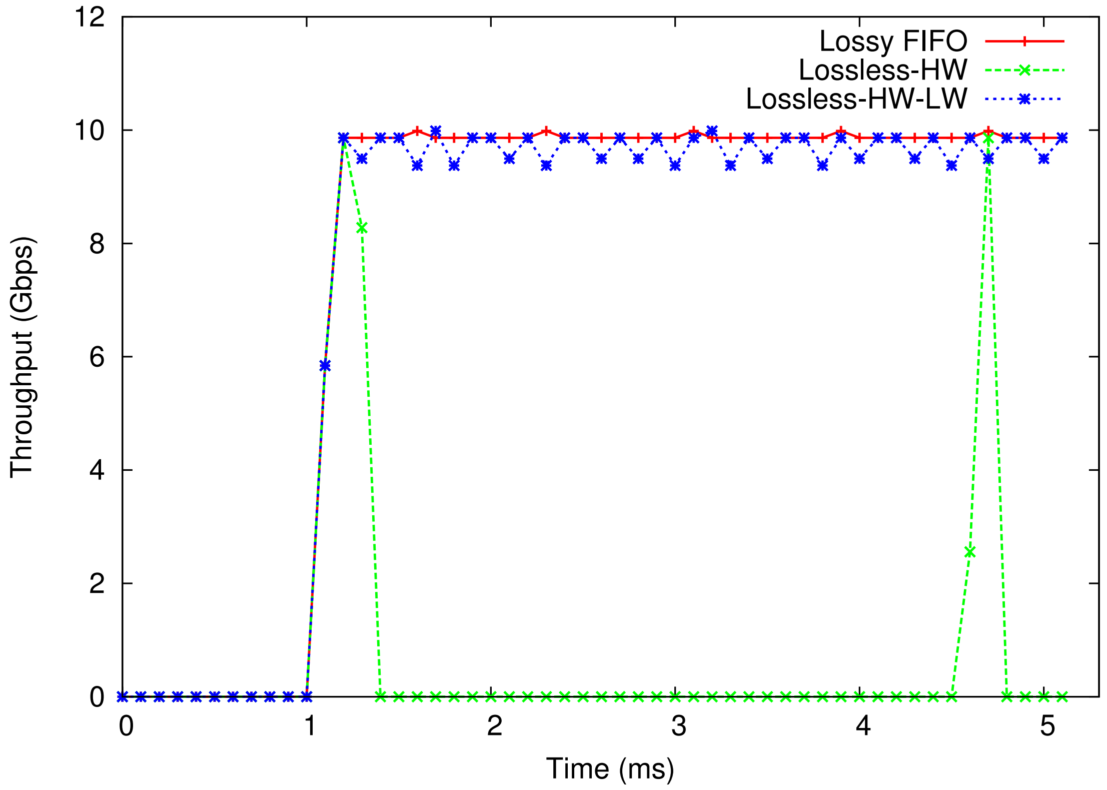


Figure 7-7: Outbound link throughput with no watermarks, with HW, and with HW and LW

With the confidence that the HW-LW mechanism can successfully maintain the utilization of the congested outbound link and remain lossless, the following experiments focus on the throughput of the inbound links. In the next experiment, a simple HW-LW System is observed (Figure 7-8). Two of the inbound traffic streams (the victim streams) start to transmit at 1 ms with a steady rate of about 2.5 Gbps (1/4 the 10 Gbps line rate). The third inbound traffic stream (the aggressor stream) starts to transmit at 1 ms with the same 2.5 Gbps rate as the two victim streams. At 3 ms the aggressor stream stops and 1 ms later it starts to transmit again at line rate, which congests the network.

While the three streams are transmitting at 1/4 line rate, it can be seen that the three streams evenly split the available bandwidth without any congestion and no pause is seen. After the aggressor stream starts to transmit at line rate the network quickly becomes congested. Although the victim streams continue to transmit at the same rate, less than their fair share of bandwidth, the throughput of all incoming traffic streams are disrupted. Each traffic stream's transmission rate has a saw-tooth pattern, in which all three incoming traffic streams transmit as fast as possible followed by stopping until the queue drains all the way to the LW.

7.4.1 Random Sampling

The same experiment was then run with the proposed random sampling mechanism implemented. The results of this experiment show that the fates of the streams start to decouple (Figure 7-9). The two victim streams are now only affected infrequently, when they are targeted by the random sampling. When a link partner transmitting a victim stream is targeted by the pause, the queue continues to grow because the aggressor stream is over-saturating the link until the HW is triggered and throttles all of the traffic. It takes a period of time to settle the streams, then the link partner with the aggressor stream is accurately targeted for a span of time again. Pseudo code for an example implementation of the Random Sampling mechanism is provided in Appendix A.

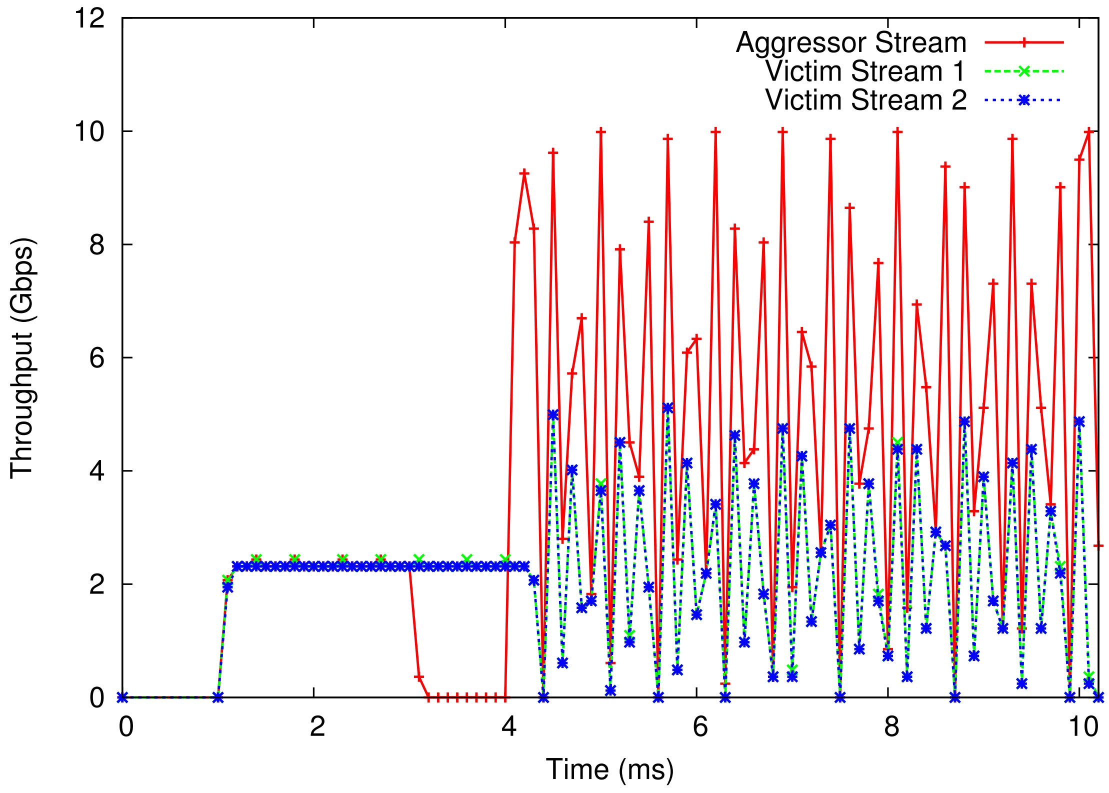


Figure 7-8: Input link throughput with HW and LW with no aggressor stream targeting

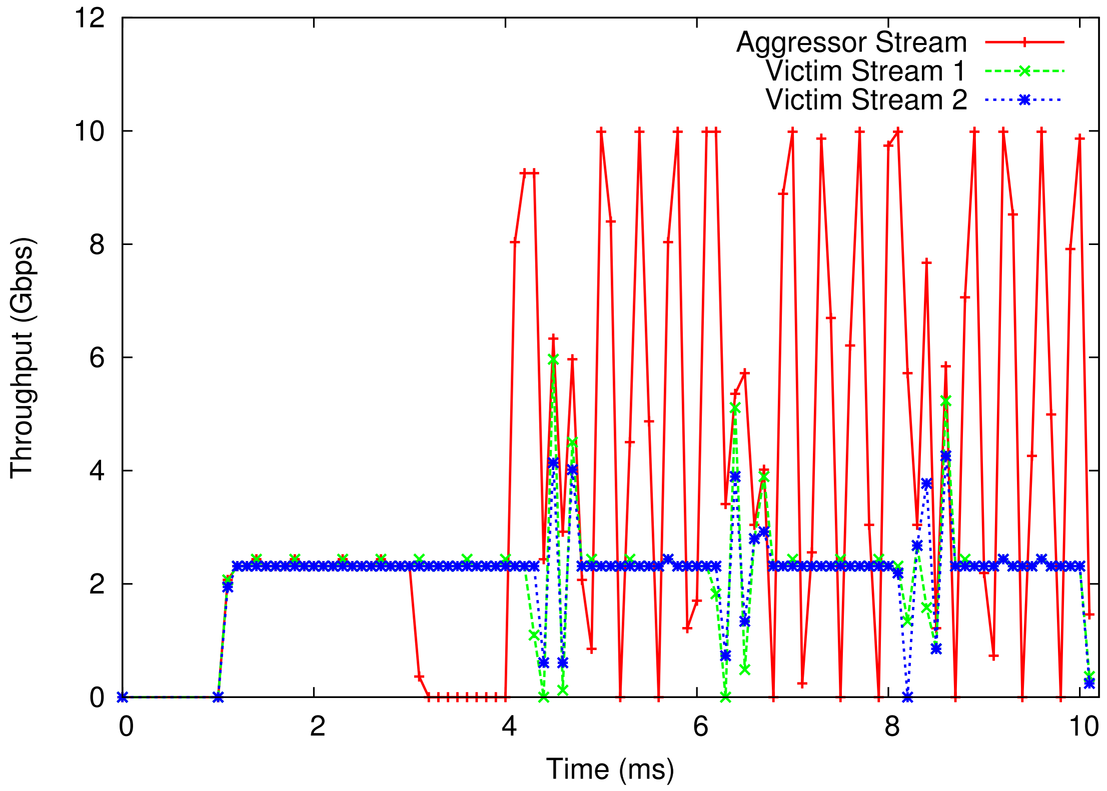


Figure 7-9: Input link throughput with random sampling targeting

7.4.2 Fair Bandwidth

The fair bandwidth mechanism targets the link partner with the aggressor stream only (Figure 7-10). There are several ways in which to implement this in a real system. Naively, an implementation could simply search the queue whenever the TW is reached and count the number of different link partners to assign the fair bandwidth. This approach would not require any additional memory, but would cost processing power and would be too slow for data center needs. Pseudo code for an example implementation of this version of the Fair Bandwidth mechanism is provided in Appendix B.

Many switches today already keep extensive statistics, so a less processor intensive implementation might keep track of the frames as they enter and leave the queue to calculate the fair bandwidth in real time. Modern switches keep running hardware statistics, including: size, priority, multicast or broadcast address, that do not require any processing power for every packet traversing the switch. It would be trivial to extend the existing statistics to monitor the number of frames in the queue from different ingress ports. This would require a small increase in memory to store the statistics, but would be fast and require minimal additional processing power.

7.4.3 Throughput standard deviation of victim stream

The standard deviation of the throughput of the victim streams is used to measure the impact of the aggressor stream on the victim streams. As the accuracy of the aggressor stream targeting improves, the standard deviation of the victim streams is decreased (Table 7.1). It can be seen that the random sampling mechanism significantly reduces the standard deviation from the baseline HW-LW experiment of 3.4 Gbps down to 0.9 Gbps. The fair bandwidth mechanism reduces the standard deviation further down to 0.5 Gbps. The standard deviation of victim stream 2 was found to be similar

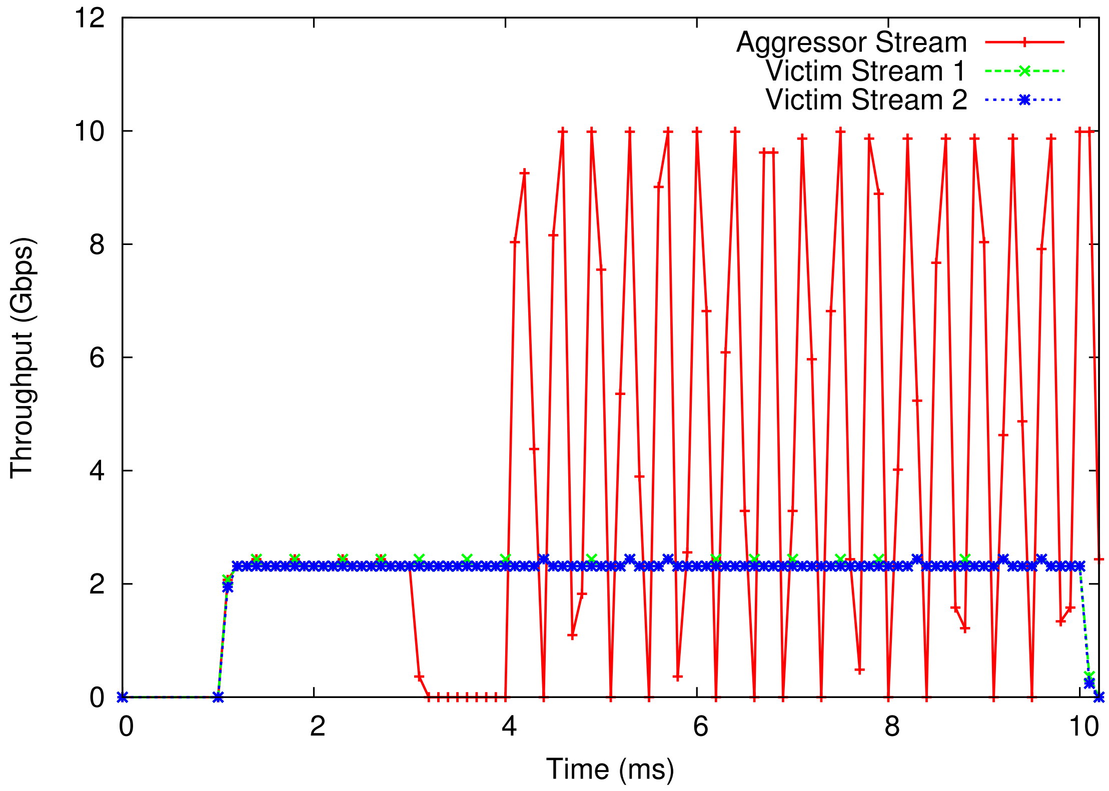


Figure 7-10: Input link throughput with fair bandwidth targeting

Table 7.1: Throughput standard deviation of victim stream 1

HW-LW	**Random Sampling**	**Fair Bandwidth**
3.4 Gbps	0.9 Gbps	0.5 Gbps

to victim stream 1 and the standard deviation of the aggressor stream is not of concern as it will purposefully increase as the aggressor stream is targeted more accurately.

7.5 Variable traffic and large topologies

TPFC builds upon well-known mechanics of Ethernet flow control consisting of IEEE 802.3x Flow Control [84] as well as its extension, IEEE 802.1Qbb Priority-based Flow Control [1]. Ethernet flow control has been shown to smooth bursty traffic [80]. Additionally, as discussed in Section 7.3, the IEEE standard [1] defines a link delay calculation that determines where to set the HW to prevent frame loss in a worst-case burst of traffic. Together, both features of Ethernet flow control handle any length burst of traffic as well as smooth out the burst across the network. TPFC then builds upon this base and minimizes fate sharing amongst traffic streams.

Since TPFC builds upon well-known Ethernet flow control mechanisms and works on a hop-by-hop basis, it will operate the same regardless of the complexity of the topology. The point of congestion in a network will send TPFC frames to its neighbors, the neighbors will then send TPFC frames to their neighbors as they become congested. In this manner, TPFC will target aggressor streams at the point of congestion and at each hop along the way back to the source of the aggressor stream, as can be seen in the example in Figure 7-11. The Ethernet flow control link layer pause mechanism has been shown in practice to scale well with network size and complexity. Since TPFC does not alter the nature of Ethernet flow control, it should scale

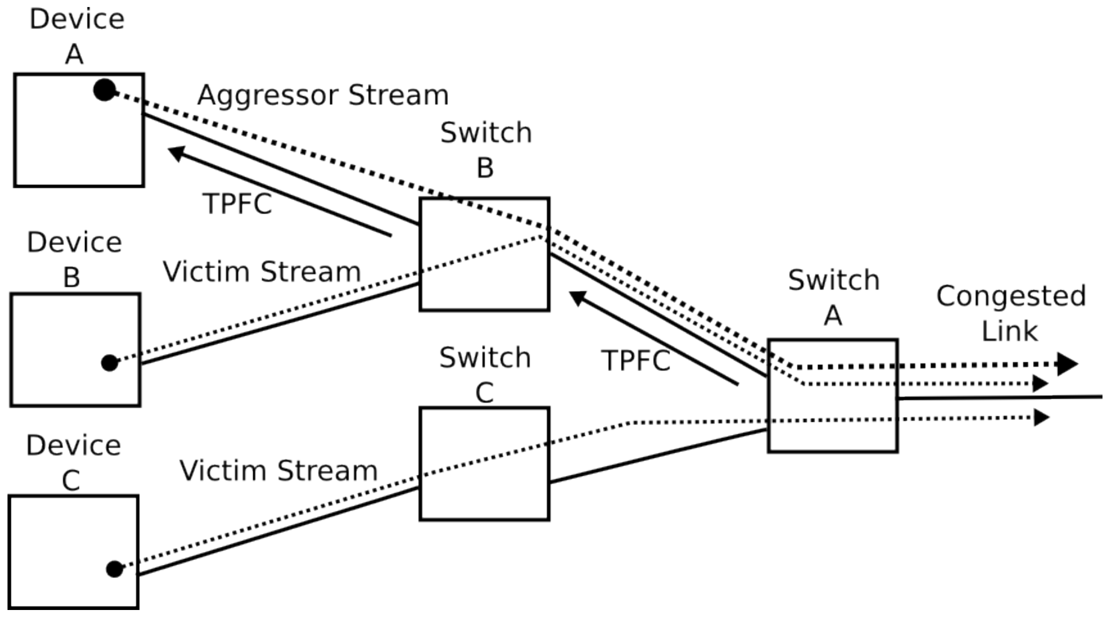


Figure 7-11: Targeted PFC over a multi-hop network: (1) When Switch A becomes congested, it sends a TPFC to Switch B only. (2) When Switch B becomes congested, it sends a TPFC to Device A.

similarly.

7.6 Interworking of TPFC and CN

As discussed in Section 7.1, CN by itself does not prevent frame loss and its effectiveness is constrained by the round trip time from the point of congestion to the source of congestion. TPFC operates on a hop-by-hop basis for a rapid local response to congestion. By combining TPFC and CN, TPFC responds quickly to bursts of congestion while CN reduces the transmission rates of aggressor stream sources.

The TPFC random sampling mechanism is similar to how QCN works. Implementations of both mechanisms can share the same watermarks. Upon reaching a watermark, the queue will be randomly sampled. Since PFC is priority-based and CN is not, TPFC frames will be sent based on sources within a priority class of sampled packets. CN congestion notification messages (CNM) will be generated based on information from all sampled packets regardless of priority class. This will pause the aggressor stream before victim streams can be disrupted and eventually the CNM will reach the source of the aggressor stream to cause it to reduce its transmission rate as well.

The TPFC fair bandwidth mechanism and CN do not share the same watermarks. As a result, CN will start to sample the queue early and send a CNM to instruct the aggressor source to reduce its transmission rate and, if a rapid burst of traffic occurs and CN is not fast enough, the watermark for TPFC will be reached and a TPFC frame will be sent in order to rapidly pause aggressor streams.

TPFC complements CN and improves its responsiveness to rapid increases in traffic load.

7.7 Conclusion

This chapter builds upon the concepts found in PFC and CN and proposes a new mechanism called TPFC that can target link partners with aggressor streams on a hop-by-hop basis in order to rapidly respond to network congestion. With the introduction of a TW below the HW within the queue, the new proposed mechanisms can target link partners with aggressor traffic streams before the HW is reached.

Performance of the proposed mechanisms were evaluated using the ns-2 simulator. An easy to implement random sampling mechanism that targets an aggressor stream with higher frequency than the victim streams showed a decoupling of stream fates. The implementation complexity was increased using a fair bandwidth mechanism in order to target only aggressor streams. Both the random sampling mechanism and the fair bandwidth mechanism showed significant reductions in the standard deviation of victim stream throughput from the base HW-LW implementation.

Chapter 8

Conclusion

Data Center Bridging (DCB) is a series of new extensions to Ethernet in order to allow Ethernet to become a better media for convergence. Converging storage, inter-processor communication and general networking into a single transmission media provides many different benefits. Some of the benefits include reducing power costs, reducing cooling and reducing the knowledge required to maintain multiple transmission media. This research began by observing that DCB significantly changes the behavior of Ethernet and it would be important to understand the performance benefits and limitations of these changes.

Chapter 3 provided further understanding beyond the background for each of the individual protocols within DCB [85]. This study examined some of the typical pitfalls involved with trying to meausure and test the different DCB protocols. Finally, it provided some interesting case studies on problems found within devices that have implemented the protocols.

Chapter 4 involved understanding what the performance implications were of converging storage traffic (iSCSI) and web traffic (TCP/UDP) over data center grade hardware without the DCB protocols enabled [78]. This study focused on converging iSCSI on a network with either 2 or 3 streams of TCP and UDP traffic, then measuring the throughput before and after being converged. It was observed that when iSCSI contends directly with TCP/UDP it suffers a larger throughput reduction than the TCP/UDP streams. Also, when iSCSI does not contend directly with TCP/UDP due to more traffic flowing in the reverse direction of the TCP/UDP traffic streams, it is better able to utilize the network and suffers less of a reduction of throughput. Finally, it was discussed how the different technologies within DCB, specifically PFC and ETS, could improve the performance of the iSCSI traffic stream.

Chapter 5 investigated actual DCB hardware and examined the performance benefits and limitations of PFC in a small network [79]. This study began by looking at PFC with generic traffic streams and was able to show a significant (about 40%) reduction in frame loss by simply enabling PFC when the receiver on a network cannot handle as much as the transmitter is sending. We also looked at the limitations of PFC when multiple streams are configured within the same priority and the effect on victim streams in that priority class when PFC starts to be seen on the network due to aggressor traffic streams in that priority class over utilizing the network. This study ended with a look at the performance of two major applications, iSCSI and MPI, when PFC is enabled and disabled on the network. We were able to show a significant throughput increase for both iSCSI write and read operations when PFC is enabled. Using MPI, we were able to show that latency, which is the key metric in MPI applications, was more consistent and on average lower when PFC is enabled. Both applications were able to benefit from avoiding the overhead associated with TCP when frame loss occurs.

Chapter 6 examined the performance effects of traditional scheduling algorithms like Deficit Round Robin (DRR) scheduling within a DCB network and proposed a new DRR scheduling algorithm to deal with one of the major performance limitations [86]. When DRR is enabled on a network where the switch includes inter-frame gap in the DRR scheduling algorithm, it was observed that small frames suffered reduced performance compared to larger frames. Utilizing a programmable switch, we were able to adjust the weights of the queues within the switch in order to overcome the throughput deficit of small frames. Using this data, we developed a new algorithm called DRR-AWC, to adjust the weights and included this in a new scheduling algorithm. We then implemented the algorithm in the programmable switch in order to test the accuracy of the algorithm. In the end we were able to show a significant improvement to the fairness of the streams with Jain's Fairness Index.

Chapter 7 began by observing the device manufacturers were not implementing CN due to its complexity. It became clear that a new algorithm should be developed that was able to provide the benefits of CN without the penalty of such complexity and slow response time. We subsequently developed a new algorithm called Targeted Priority-based Flow Control (TPFC) [87]. Using simulations, we were able to develop two different mechanisms to target aggressor streams and provide a significant increase in the fairness of the network.

This research serves as a starting point for much more research that needs to be done on DCB. Priority-based Flow Control, Enhanced Transmission Selection and Congestion Notification need to continue to be researched individually and together. It will be interesting to see the performance implications of a network when all three protocols are enabled at the same time. Will the system work as planned? Are there any surprising interactions? Finally, applications have been developed over the past decades with the assumption that frame loss will occur during congestion, so how do applications need to change if this basic assumption is eliminated?

Bibliography

[1] "IEEE Standard for Local and Metropolitan Area Networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks - Amendment 17: Priority-based Flow Control," *IEEE Std 802.1Qbb-2011*, 2011.

[2] "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Amendment 8: MAC Control Frame for Priority-based Flow Control," *IEEE Std 802.3bd-2011*, 2011.

[3] "Weighted fair queuing diagram." `http://www.h3c.com/`.

[4] M. Hagen and E. Varki, "Fibre channel arbitrated loop-performance analysis of loop tenancy overhead," in *Networking, Architecture, and Storage, 2008. NAS'08. International Conference on*, pp. 225–232, IEEE, 2008.

[5] O. Feuser and A. Wenzel, "On the effects of the IEEE 802.3x flow control in full-duplex ethernet lans," in *Local Computer Networks, 1999. LCN'99. Conference on*, pp. 160–161, IEEE, 1999.

[6] A. Odlyzko, "Internet traffic growth: Sources and implications," in *ITCom 2003*, pp. 1–15, International Society for Optics and Photonics, 2003.

[7] C. Bestler and L. Coene, "Applicability of Remote Direct Memory Access Protocol (RDMA) and Direct Data Placement (DDP)," tech. rep., RFC 5045, October, 2007.

[8] J. Pinkerton and E. Deleganes, "Direct Data Placement Protocol (DDP)/Remote Direct Memory Access Protocol (RDMAP) Security," *Self*, 2007.

[9] R. Recio, P. Culley, B. Metzler, D. Garcia, and J. Hilland, "A remote direct memory access protocol specification," 2007.

[10] A. Romanow, J. Mogul, T. Talpey, and S. Bailey, "Remote Direct Memory Access (RDMA) over IP Problem Statement," *The Internet Society*, 2005.

[11] S. Bailey and T. Talpey, "The architecture of direct data placement (ddp) and remote direct memory access (rdma) on internet protocols," *Architecture*, 2005.

[12] H. Shah, J. Pinkerton, R. Recio, and P. Culley, "Direct data placement over reliable transports," tech. rep., IETF Internet-draft draft-ietf-rddp-ddp-04. txt (work in progress), 2005.

[13] P. Culley, U. Elzur, R. Recio, S. Bailey, and J. Carrier, "Marker PDU aligned framing for TCP specification," tech. rep., IETF Internet-draft draft-ietf-rddp-mpa-02. txt (work in progress), 2005.

[14] D. Dalessandro, P. Wyckoff, and G. Montry, "Initial performance evaluation of the NetEffect 10 Gigabit iWARP adapter," in *Cluster Computing, 2006 IEEE International Conference on*, pp. 1–7, IEEE, 2006.

[15] "Fibre Channel Backbone-5 (FC-BB-5) Rev 2.00," tech. rep., T11/Project 1871-D/Rev 2.00, 2009.

[16] M. Disabato, "Converged enhanced ethernet: Weaving the unified fabric," 2009.

[17] R. Zarick, M. Hagen, and R. Bartos, "Transparent clocks vs. enterprise ethernet switches," in *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2011 International IEEE Symposium on*, pp. 62–68, IEEE, 2011.

[18] R. Zarick, M. Hagen, and R. Bartos, "The impact of network latency on the synchronization of real-world ieee 1588-2008 devices," in *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2010 International IEEE Symposium on*, pp. 135–140, IEEE, 2010.

[19] "IEEE Standard for Local and Metropolitan Area Networks - Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks - Amendment 18: Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Class," *IEEE Std 802.1Qaz-2011*, 2011.

[20] "IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks - Amendment 13: Congestion Notification," *IEEE Std 802.1Qau-2010*, 2010.

[21] "IEEE Standard for Local and Metropolitan Area Networks - Station and Media Access Control Connectivity Discovery," *IEEE Std 802.1AB-2009*, 2009.

[22] M. Karaata and P. Chaudhuri, "A self-stabilizing algorithm for strong fairness," *Computing*, vol. 60, no. 3, pp. 217–228, 1998.

[23] L. Wischik, L. Lamport, and Y. Yu, "How to verify temporal formulas in TLA." `http://www.wischik.com/lu/research/verify-tla/talk-paged.html`.

[24] T. Nguyen and Y. Han, "A proportional fairness algorithm with QoS provision in downlink OFDMA systems," *Communications Letters, IEEE*, vol. 10, no. 11, pp. 760–762, 2006.

[25] S. Gjessing and A. Maus, “A fairness algorithm for high-speed networks based on a resilient packet ring architecture,” in *Proceedings of 2002 IEEE International Conference on Systems, Man and Cybernetics*, vol. 2, pp. 279–284, 2002.

[26] I. Cidon and Y. Ofek, “Metaring-a full-duplex ring with fairness and spatial reuse,” *Communications, IEEE Transactions on*, vol. 41, no. 1, pp. 110–120, 1993.

[27] J. Chen, I. Cidon, and Y. Ofek, “A local fairness algorithm for gigabit LAN’s/MAN’s with spatial reuse,” *Selected Areas in Communications, IEEE Journal on*, vol. 11, no. 8, pp. 1183–1192, 1993.

[28] J. Chen, I. Cidon, and Y. Ofek, “A local fairness algorithm for the metaring, and its performance study,” in *Global Telecommunications Conference, 1992. Conference Record., GLOBECOM’92. Communication for Global Users., IEEE*, pp. 1635–1641, IEEE, 1992.

[29] X. Zhou, G. Shi, H. Fang, and L. Zeng, “Fairness algorithm analysis in resilient packet ring,” in *Communication Technology Proceedings, 2003. ICCT 2003. International Conference on*, vol. 1, pp. 622–624, IEEE, 2003.

[30] J. Le Boudec, “Rate adaptation, congestion control and fairness: a tutorial,” *Web page, November*, 2005.

[31] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, “Achieving MAC layer fairness in wireless packet networks,” in *Proceedings of the 6th annual international conference on Mobile computing and networking*, pp. 87–98, ACM, 2000.

[32] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, “MACAW: a media access protocol for wireless LAN’s,” in *ACM SIGCOMM Computer Communication Review*, vol. 24, pp. 212–225, ACM, 1994.

[33] T. Ozugur, M. Naghshineh, P. Kermani, C. Olsen, B. Rezvani, and J. Copeland, "Balanced media access methods for wireless networks," in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 21–32, ACM, 1998.

[34] A. Mayer, Y. Ofek, and M. Yung, "Local and congestion-driven fairness algorithm in arbitrary topology networks," *Networking, IEEE/ACM Transactions on*, vol. 8, no. 3, pp. 362–372, 2000.

[35] Y. Ofek, "Overview of the MetaRing architecture," *Computer Networks and ISDN Systems*, vol. 26, no. 6, pp. 817–829, 1994.

[36] D. Yin and J. Xie, "Probability based weighted fair queueing algorithm with adaptive buffer management for high-speed network," *Advances in Natural Computation*, pp. 428–437, 2006.

[37] R. El Khoury and R. El-Azouzi, "Modeling the effect of forwarding in a multi-hop ad hoc networks with weighted fair queueing," *Mobile Ad-Hoc and Sensor Networks*, pp. 5–18, 2007.

[38] J. Georges, T. Divoux, and E. Rondeau, "Strict priority versus weighted fair queueing in switched ethernet networks for time critical applications," in *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, pp. 141–141, IEEE, 2005.

[39] L. Wang, D. Fayek, and T. Sivananthan, "A bandwidth bargain model based on adaptive weighted fair queueing," in *Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP*, pp. 1–4, IEEE, 2006.

[40] T. J. Kim, “NXG01-2: a weighted fair queuing with optimal rate and delay allocation,” in *Global Telecommunications Conference, 2006. GLOBECOM ’06. IEEE*, pp. 1 –4, 27 2006-dec. 1 2006.

[41] M. Fischer, D. Masi, and J. Shortle, “Simulating the performance of a class-based weighted fair queueing system,” in *Proceedings of the 40th Conference on Winter Simulation*, pp. 2901–2908, Winter Simulation Conference, 2008.

[42] K. Khawam and D. Kofman, “Opportunistic weighted fair queueing,” in *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, pp. 1–5, IEEE, 2006.

[43] H. Luo, J. Cheng, and S. Lu, “Self-coordinating localized fair queueing in wireless ad hoc networks,” *Mobile Computing, IEEE Transactions on*, vol. 3, no. 1, pp. 86–98, 2004.

[44] D. Masi, M. Fischer, and D. Garbin, “Modeling the performance of low latency queueing for emergency telecommunications,” in *Proceedings of the 39th conference on Winter simulation*, vol. 40, pp. 09–12, 2007.

[45] D. Abendroth, M. Eckel, and U. Killat, “Solving the trade-off between fairness and throughput: token bucket and leaky bucket-based weighted fair queueing schedulers,” *AEU-International Journal of Electronics and Communications*, vol. 60, no. 5, pp. 404–407, 2006.

[46] A. Demers, S. Keshav, and S. Shenker, “Analysis and simulation of a fair queueing algorithm,” in *ACM SIGCOMM Computer Communication Review*, vol. 19, pp. 1–12, ACM, 1989.

[47] M. Shreedhar and G. Varghese, “Efficient fair queuing using deficit round-robin,” *Networking, IEEE/ACM Transactions on*, vol. 4, no. 3, pp. 375–385, 1996.

[48] B. Phillips, "Have storage area networks come of age?," *Computer*, vol. 31, no. 7, pp. 10–12, 1998.

[49] X. Molero, F. Silla, V. Santonja, and J. Duato, "Modeling and simulation of storage area networks," in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2000. Proceedings. 8th International Symposium on*, pp. 307–314, IEEE, 2000.

[50] R. Telikepalli, T. Drwiega, and J. Yan, "Storage area network extension solutions and their performance assessment," *Communications Magazine, IEEE*, vol. 42, no. 4, pp. 56–63, 2004.

[51] J. Heath and P. Yakutis, "High speed storage area networks using a fibre channel arbitrated loop interconnect," *Network, IEEE*, vol. 14, no. 2, pp. 51–56, 2000.

[52] X. He, Q. Yang, and M. Zhang, "A caching strategy to improve iSCSI performance," in *Local Computer Networks, 2002. Proceedings. LCN 2002. 27th Annual IEEE Conference on*, pp. 278–285, IEEE, 2002.

[53] S. Aiken, D. Grunwald, A. Pleszkun, and J. Willeke, "A performance analysis of the iSCSI protocol," in *Mass Storage Systems and Technologies, 2003.(MSST 2003). Proceedings. 20th IEEE/11th NASA Goddard Conference on*, pp. 123–134, IEEE, 2003.

[54] "University of New Hampshire InterOperability Laboratory - DCB Consortium Knowledge Base." `http://www.iol.unh.edu/services/testing/dcb/training/`.

[55] Y. Adam, B. Fillinger, I. Astic, A. Lahmadi, and P. Brigant, "Deployment and test of IPv6 services in the VTHD network," *Communications Magazine, IEEE*, vol. 42, no. 1, pp. 98–104, 2004.

[56] "IPv6 Ready Logo Program Approved List." `https://www.ipv6ready.org/db/index.php/public/`.

[57] A. Vallejo, J. Ruiz, J. Abella, A. Zaballos, and J. Selga, "State of the art of IPv6 conformance and interoperability testing," *Communications Magazine, IEEE*, vol. 45, no. 10, pp. 140–146, 2007.

[58] P. Henry and H. Luo, "WiFi: what's next?," *Communications Magazine, IEEE*, vol. 40, no. 12, pp. 66–72, 2002.

[59] P. De, A. Raniwala, S. Sharma, and T. Chiueh, "Design considerations for a multihop wireless network testbed," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 102–109, 2005.

[60] R. Rao, W. Zhu, S. Lang, C. Oberli, D. Browne, J. Bhatia, J. Frigon, J. Wang, P. Gupta, H. Lee, *et al.*, "Multi-antenna testbeds for research and education in wireless communications," *Communications Magazine, IEEE*, vol. 42, no. 12, pp. 72–81, 2004.

[61] "University of New Hampshire InterOperability Laboratory - DCB Consortium Test Suites." `http://www.iol.unh.edu/services/testing/dcb/testsuites.php`.

[62] "University of New Hampshire InterOperability Laboratory - iSCSI over DCB Integrator's List." `http://www.iol.unh.edu/services/testing/dcb/list.php`.

[63] K. Leigh, P. Ranganathan, and J. Subhlok, "Fabric convergence implications on systems architecture," in *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pp. 15–26, IEEE, 2008.

[64] Y. Lu and D. Du, "Performance study of iSCSI-based storage subsystems," *Communications Magazine, IEEE*, vol. 41, no. 8, pp. 76–82, 2003.

[65] H. Simitci, C. Malakapalli, and V. Gunturu, "Evaluation of SCSP over TCP/IP and SCSI over fibre channel connections," in *Hot Interconnects 9, 2001.*, pp. 87–91, IEEE, 2001.

[66] H. Khosravi, A. Joglekar, and R. Iyer, "Performance Characterization of iSCSI processing in a server platform," in *Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International*, pp. 99–107, IEEE, 2005.

[67] "Open MPI: Open Source High Performance Computing." `http://www.open-mpi.org`.

[68] "Intel MPI Benchmarks." `http://software.intel.com/en-us/articles/intel-mpi-benchmarks/`.

[69] "IEEE 802.1 Data Center Bridging Task Group." `http://www.ieee802.org/1/pages/dcbridges.html`.

[70] "UNH iSCSI Project." `http://unh-iscsi.svn.sourceforge.net/`.

[71] E. Laias, I. Awan, and P. Chan, "Fair and latency aware uplink scheduler in IEEE 802.16 using customized deficit round robin," in *Advanced Information Networking and Applications Workshops, 2009. WAINA'09. International Conference on*, pp. 425–432, IEEE, 2009.

[72] H. Chaskar and U. Madhow, "Fair scheduling with tunable latency: a round-robin approach," *IEEE/ACM Transactions on Networking (TON)*, vol. 11, no. 4, pp. 592–601, 2003.

[73] K. Yamakoshi, K. Nakai, E. Oki, and N. Yamanaka, "Dynamic deficit round-robin scheduling scheme for variable-length packets," *Electronics Letters*, vol. 38, no. 3, pp. 148–149, 2002.

[74] C. So-In, R. Jain, and A. Tamimi, “A deficit round robin with fragmentation scheduler for IEEE 802.16e mobile WiMAX,” in *Sarnoff Symposium, 2009. SARNOFF’09. IEEE*, pp. 1–7, IEEE, 2009.

[75] K. Thompson, G. Miller, and R. Wilder, “Wide-area internet traffic patterns and characteristics,” *Network, IEEE*, vol. 11, no. 6, pp. 10–23, 1997.

[76] N. Brownlee and K. Claffy, “Understanding internet traffic streams: dragonflies and tortoises,” *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 110–117, 2002.

[77] R. Jain, D. Chiu, and W. Hawe, *A quantitative measure of fairness and discrimination for resource allocation in shared computer system.* Eastern Research Laboratory, Digital Equipment Corporation, 1984.

[78] M. Hagen and E. Varki, “iSCSI on a converged data center network,” in *22nd International Conference on Parallel and Distributed Computing and Communication Systems (PDCCS)*, pp. 97–102, ISCA, 2009.

[79] M. Hagen and R. Zarick, “Performance evaluation of DCB’s priority-based flow control,” in *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, pp. 328–333, IEEE, 2011.

[80] M. Hayasaka, T. Sekiyama, S. Oshima, and T. Sakuraba, “Dynamic pause time calculation method in MAC layer flow control,” in *Broadband Multimedia Systems and Broadcasting (BMSB), 2010 IEEE International Symposium on*, pp. 1–5, IEEE, 2010.

[81] R. Pan, B. Prabhakar, and A. Laxmikantha, “QCN: Quanitized Congestion Notification.” `http://www.stanford.edu/~balaji/presentations/au-prabhakar-qcn-description.pdf`.

[82] Y. Hayashi, H. Itsumi, and M. Yamamoto, "Improving fairness of quantized congestion notification for data center ethernet networks," in *Distributed Computing Systems Workshops (ICDCSW), 2011 31st International Conference on*, pp. 20–25, IEEE, 2011.

[83] "The Network Simulator — ns-2." `http://nsnam.isi.edu/nsnam/index.php/User_Information`.

[84] "IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications," *IEEE Std 802.3-2008 (Revision of IEEE Std 802.3-2005)*, 2008.

[85] M. Hagen, P. Scruton, R. Noseworthy, R. Zarick, and R. Bartos, "Testing challenges of data center bridging networks," *Communications Magazine, IEEE*, vol. 50, no. 3, pp. 140–145, 2012.

[86] M. Hagen, R. Zarick, and R. Bartos, "Deficit round robin scheduling with adaptive weight control," in *25th International Conference on Computer Applications In Industry And Engineering (CAINE)*, ISCA, 2012.

[87] M. Hagen, R. Bartos, J. Fastabend, and R. Noseworthy, "Targeted priority-based flow control," *Under review at IEEE Communications Letters*, 2012.

Appendix A

Random Sampling

Algorithm 2 Enque Function

```
if queueLength+1 $\geq$ queueLimit then
  dropPacket
else
  if queueLength $\geq$ High Watermark then
    send PFC to all ports
    enquePacket
  else if queueLength $\geq$ Target Watermark then
    randomly sample Packet P from the queue
    send PFC to the source of Packet P
    enquePacket
  else
    enquePacket
  end if
end if
```

Algorithm 3 Deque Function

```
if queueLength $\leq$ Low Watermark then
  for each port do
    sendPFC 0
  end for
end if
if not paused then
  dequePacket
end if
```

Appendix B

Fair Bandwidth

Algorithm 4 Enque Function

```
if queueLength+1 $\geq$ queueLimit then
    dropPacket
else
    if queueLength $\geq$ High Watermark then
        send PFC to all ports
        enquePacket
    else if queueLength $\geq$ Target Watermark then
        N = the number of different sources in the queue
        b = the number of frames from one source in the queue
        send PFC to each source that exceeds $b_i = \frac{b}{N}$
        enquePacket
    else
        enquePacket
    end if
end if
```

Algorithm 5 Deque Function

```
if queueLength $\leq$ Low Watermark then
    for each port do
        sendPFC 0
    end for
end if
if not paused then
    dequePacket
end if
```