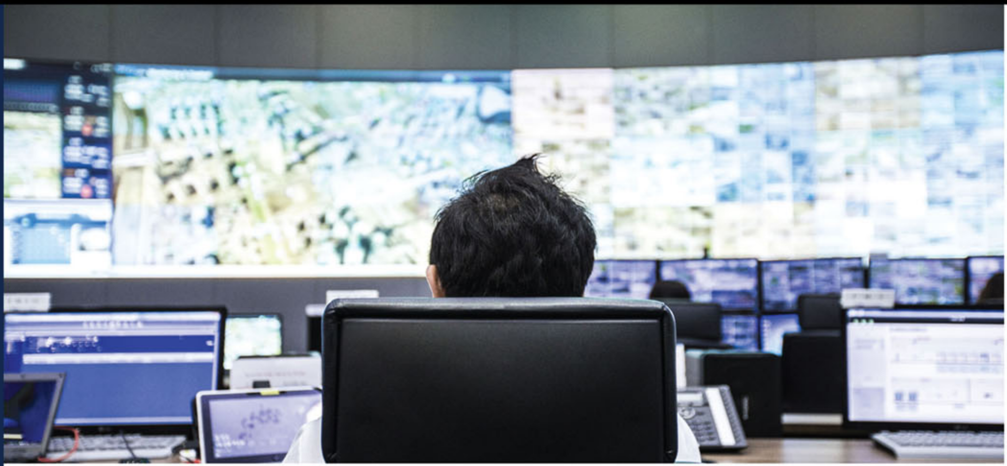

Detecting, Troubleshooting, and Preventing Congestion in Storage Networks

PARESH GUPTA, CCIE® NO. 36645
EDWARD MAZUREK, CCIE® NO. 6448

ciscopress.com

FREE SAMPLE CHAPTER

Detecting, Troubleshooting, and Preventing Congestion in Storage Networks

Paresh Gupta

Edward Mazurek

Cisco Press

Hoboken, New Jersey

Detecting, Troubleshooting, and Preventing Congestion in Storage Networks

Paresh Gupta
Edward Mazurek

Copyright© 2024 Cisco Systems, Inc.

Published by:
Cisco Press

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms, and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions.

No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

Please contact us with concerns about any potential bias at pearson.com/report-bias.html.

$PrintCode

Library of Congress Control Number: 2023920453

ISBN-13: 978-0-13-788723-1

ISBN-10: 0-13-788723-X

Warning and Disclaimer

This book is designed to provide information about detecting, troubleshooting, and preventing congestion in storage networks. Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied.

The information is provided on an "as is" basis. The authors, Cisco Press, and Cisco Systems, Inc. shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the authors and are not necessarily those of Cisco Systems, Inc.

Trademark Acknowledgments

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Cisco Press or Cisco Systems, Inc. cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Feedback Information

At Cisco Press, our goal is to create in-depth technical books of the highest quality and value. Each book is crafted with care and precision, undergoing rigorous development that involves the unique expertise of members from the professional technical community.

Readers' feedback is a natural continuation of this process. If you have any comments regarding how we could improve the quality of this book, or otherwise alter it to better suit your needs, you can contact us through email at feedback@ciscopress.com. Please make sure to include the book title and ISBN in your message.

We greatly appreciate your assistance.

Vice President, IT Professional: Mark Taub

Alliances Manager, Cisco Press: Caroline Antonio

Director, ITP Product Management: Brett Bartow

Executive Editor: James Manly

Managing Editor: Sandra Schroeder

Development Editor: Ellie Bru

Senior Project Editor: Mandie Frank

Copy Editor: Kitty Wilson

Technical Editors: Harsha Bharadwaj, Erik Smith, Fausto S. Vaninetti

Editorial Assistant: Cindy Teeters

Designer: Chuti Prasertsith

Composition: codeMantra

Indexer: Ken Johnson

Proofreader: Jennifer Hinchliffe

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at **www.cisco.com/go/offices.**

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

About the Authors

Paresh Gupta, CCIE No. 36645, has almost two decades of experience in the computer industry. Currently, as a senior leader of Technical Marketing Engineering for Cisco, he drives the technical and market evolution of products, technologies, and solutions such as SAN Analytics, Nexus Dashboard, UCS, MDS, and Nexus switches. He has been testing and validating congestion in storage networks for many years. In his multiple roles, he has invented/patented many ideas, developed many features, and trained thousands of people in sales, partner, and customer communities. Paresh is the creator of the full-blown traffic monitoring apps for Cisco UCS Servers (UTM) and MDS switches (MTM). Hundreds of organizations use Paresh's apps in production around the world.

Edward Mazurek, CCIE No. 6448, has more than 40 years of experience in the computer networking industry. The first 18 were with IBM, supporting products such as Virtual Machine (VM) and VTAM. He has spent the last 22+ years with the Cisco TAC. As a principal engineer, he supports data center networking technologies, including storage-area networking, Fibre Channel, and FCoE in the MDS, UCS, and Nexus 5000, 6000, 7000, and 9000 series. He holds two CCIEs—SNA/IP Integration (2000) and Storage Area Networking (SAN)—and is presently a CCIE Emeritus. Ed has spearheaded the congestion-handling mechanisms on the Cisco MDS 9000 switches and Nexus 9000 switches. Based on his deep understanding, Ed developed an app that is being used by other Cisco engineers and continues to be in high demand by various partners. In addition to inventing multiple features on Cisco products, Ed holds multiple patents in the field of network congestion.

About the Technical Reviewers

Harsha Bharadwaj is a distinguished engineer in the Data Center group at Cisco India. He started his career as a software developer with Andiamo Systems, which developed the MDS series of Fibre Channel switches and was later acquired by Cisco. His later stints have included designing and developing protocol and platform software across the Cisco data center portfolio of Nexus 7000, Nexus 5000, and MDS switches. He has spent about 23 years at Cisco. In recent years, his focus has been on the architectural aspects of MDS switches and keeping track of trends in storage and storage networking. He holds about 20 patents in these areas. He also represents Cisco at T11 (INCITS Technical Committee for Fibre Channel Standards) and UEC (Ultra Ethernet Consortium).

Erik Smith is a distinguished engineer, currently working for Dell's Chief Technology and Innovation Office (CTIO). During the past 25 years, Erik has primarily focused on storage-area networks (SANs) and the protocols associated with them, including traditional Fibre Channel, FCoE, iSCSI, NVMe/FC, and NVMe/TCP. Erik was an active member of INCITS T11 (the FC standards group), especially during the creation of the first FCoE standard (FC-BB-5), and the NVM Express FMDS working group. He is currently the vice chair of SNIA's Network Storage Forum. Erik is the inventor of multiple technologies, including target-driven zoning (TDZ), virtual storage networks, and zero-touch infrastructure provisioning, and he has more than 50 related patents (granted and pending).

Fausto S. Vaninetti is a senior solutions engineer for Data Center Architectures at Cisco Systems. With more than 30 years' experience in the ICT sector, his primary focus has expanded from data center optical interconnection and storage networking to cloud computing and sustainability. He is the go-to person for technical presales on storage-related topics and has been a speaker at several events, including Cisco Live, EMC World, IBM STGU, and the Storage Developer Conference. For a few years he had a position on the board of directors of SNIA Europe. He has authored a significant number of papers and blog posts and coauthored a book on FICON technology. At present, he is the reference point in Cisco EMEA for data center sustainability evangelization. He is based in Cisco's Milan, Italy office.

Dedications

Paresh Gupta

I dedicate this book to my wife, Dimple. Her fun, lively, and honest personality has been a blessing in my life.

I also dedicate this book to Kiara, the best girl ever, and Manan, the best boy ever. Behind the writing of this book lie these conversations:

Kiara: “Daddy is always working.”

Manan: “Kiara, ssshhh, don’t disturb. Daddy is writing a book.”

I dedicate this book also to my late mother (Chandresh), who made me who I am today, my sisters (Renu and Sapna), who shaped me into who I am today, and my uncle (Ashok), who stood by me through the ups and downs of life.

Edward Mazurek

I dedicate this book to my wife, Bobbi Ann. Her support for over 40 years has enabled me to focus on my career, taking on multiple technical roles at both IBM and Cisco. Her love for me and our family and her ability to manage the home and our three children, Julie Ann, Alexander, and Joshua, has freed me to take advantage of many opportunities to excel. This has enabled me to be an expert in multiple technologies, eventually culminating in storage networking. She has been very tolerant of the many technical conversations she has overheard about congestion in storage networks, only closing the door to my home office occasionally.

I’d also like to dedicate this book to my late father, Robert Mazurek, who always encouraged and supported me along the way. He loved hearing about my accomplishments and endeavors whenever we got together.

Acknowledgments

Paresh Gupta

I want to thank my coauthor, Ed. It has been an absolute pleasure to work with him on this and many other projects.

A big thanks to Anshul Tanwar and Lukas Krattiger for their guidance in getting me started in writing this book.

I'd like to give a special acknowledgment to my management team at Cisco—Yousuf Khan, Andy Sholomon, and Kamal Bakshi—for creating a culture that has encouraged me and an environment that has allowed me to write this book.

A special thanks to Mark Allen for being a motivational mentor and a wonderful person.

Edward Mazurek

I would like to personally thank my coauthor, Paresh, for initiating this project and overcoming my initial reluctance to become involved. Without Paresh's enthusiasm, persistence, and hard work, this book would not have been possible.

Combined

We are very grateful to our technical editors: Erik Smith, Harsha Bharadwaj, and Fausto Vaninetti. Their review has made a significant improvement in this book. Besides improving the technical accuracy, they helped to make it more expansive and coherent.

We also want to thank Cisco engineers for answering our questions and clarifying the implementation of Cisco MDS, Nexus, and UCS product lines. Thanks to Sunil Varghese, Jhaanaki Krishnan, Amit Kumar, Deno Mathew, Lijo Vadakel John, Jayaprakash Nallapalingu, Suman Pasupuleti, Gubbala V. V. Krishna Rao, Bhargavi Rajagopal, Rithesh Iyer, Ravikumar Munirathnam Shetty, Harsh Patel, Shilpa Kothapalli, Suvidh Mathur, Prasanna Dharama Muruganandan, Reese Faucette, April Yu, Sean Wang, Sonu Kumar Khandelwal, Faraz Taifehesmatian, Nemanja Kamenica, Matthias Wessendorf, Frank Wang, and others.

We are grateful to the Cisco Press/Pearson publishing team—James Manly, Eleanor Bru, Mandie Frank, Kitty Wilson, Jennifer Hinchliffe, and others—for giving us this platform and helping us throughout the journey of writing this book.

Finally, we want to thank all the customers who have run into congestion issues in their production environments and allowed us to develop features, tools, and methodologies for handling the problems and collaborating on the ideas.

Contents at a Glance

Reader Services

Register your copy at www.ciscopress.com/title/ISBN for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to www.ciscopress.com/register and log in or create an account*. Enter the product ISBN 9780137887231 and click Submit. When the process is complete, you will find any available bonus content under Registered Products.

*Be sure to check the box indicating that you would like to hear from us to receive exclusive discounts on future editions of this product.

Contents

Command Syntax Conventions

The conventions used to present command syntax in this book are the same conventions used in the IOS Command Reference. The Command Reference describes these conventions as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([ ]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

Foreword

Storage infrastructure has changed considerably in the past few years due to the adoption of all-flash storage through technologies like NVMe and NVMe over Fabrics (NVMe-oF), which have made storage devices even faster. Today, performance at millions of input/output per second (IOPS), response times in microseconds, and throughput of hundreds of gigabytes per second are the new norm. Together with demanding applications for AI/ML use cases, 5G connectivity, and business-critical transactional workloads, this ultra-fast storage can transfer huge amounts of data with requirements for much lower response times.

These realities are stress-testing existing storage networks running older and lower-performance technologies such as rotational disks. The confluence of old and new also has the potential to increase the likelihood of network congestion and prevent the full capabilities of the newer technologies from being realized. Congestion has been a key concern for storage networks around the globe. At Cisco, we have seen this to be the top reason customers open support tickets and raise concerns in various other forms.

Paresh and Ed have been at the forefront of dealing with these issues. They have helped hundreds of customers design their storage networks with features resulting in congestion prevention and improved detection. They hold several patents in this field and have developed tools for congestion detection that are used by hundreds of customers and Cisco engineers alike. They are active storage technology evangelists who speak frequently at industry events and hold the distinction of being Cisco Live Distinguished Speakers. They have traveled worldwide to train customers and partners on this topic.

I'm excited that they chose to write about their first-hand experience with network congestion. A few things clearly stand out to me in this book.

It covers basic topics like flow control and goes deeper into advanced subjects like troubleshooting and prevention methods. This approach makes the book useful for newer users as well as experts.

This book covers many commonly used transports for storage networks, including Fibre Channel, TCP over lossy Ethernet, and RDMA over converged (lossless) Ethernet (RoCE). This provides a consistent approach to users, regardless of the transport under use, and facilitates learning.

The most unique aspect of this book is the multiple case studies that not only cover various real-world situations but also show step-by-step demonstrations of handling congestion issues.

The education in this book is one of a kind and will benefit users for years to come.

Yousuf Khan
Vice President, Technical Marketing, Cisco

Introduction

Congestion is perhaps the most critical problem in storage networks around the world. Over the years, we have worked with thousands of users to help them in detecting, troubleshooting, and preventing congestion in storage networks. Our common observation is that most users lack a thorough understanding of this subject, and honestly, there is not much educational content that explains this subject practically. On one end, application developers are assuming unlimited access to storage, and the underlying infrastructure details are less relevant to them. On the other end are storage infrastructure teams that handle storage management and allocation. In between, the network teams deal with connectivity with limited visibility in the I/O operations flowing through the network. This lack of awareness results in delayed detection and solution. Early congestion symptoms are often ignored until application performance is severely degraded, leading to loss of revenue for organizations and long working hours for administrators.

However, eliminating congestion completely may not be worth the effort in most production networks. A more realistic aim, however, should be to reduce the severity of congestion so that application performance is acceptable.

Any network that is being used for reading data and writing data to a remote storage device is a storage network for the purposes of this book. Remote storage can be inside a SAN storage array, NAS device, public cloud, or even commodity servers being used with a software-defined storage (SDS) solution or a distributed file system such as Hadoop Distributed File System (HDFS).

The impact of congestion in these networks is much more severe than in a general-purpose network because applications can't proceed if data access is slow. Although reducing or eliminating congestion in storage networks has always been a top priority, in the past decade, the massive increase in data, together with the wide adoption of all-flash storage, has made congestion even more prominent in data centers around the world. In addition, newer technologies like NVMe and NVMe over Fabrics are expected to increase network utilization to unprecedented levels.

Congestion in storage networks goes by many names. Fibre Channel users typically call it *slow drain*, even though, as you will see, this term covers merely a subset of the problems. In lossless Ethernet networks, the term *PFC storm* has emerged in the past few years, essentially referring to the same phenomenon. Among the TCP/IP networking community, TCP's built-in flow-control and congestion-control mechanisms are well known. This book explains all these concepts and explains their relevance for storage traffic. More importantly, the focus is on the actions that users can take to detect, troubleshoot, and prevent congestion in storage networks.

Who This Book Is For

In addition to explaining how technology works, this book focuses on the practical use of technology. It takes a practical approach to solving problems within the constraints and challenges of the real world, where "just upgrade" is not a solution or at least cannot be applied quickly. It explains why some solutions, despite being technically viable, cannot be applied because they don't align with the business or operations goals. We wrote this book for users of the technology, products, and solutions. At Cisco, we call them customers. In particular, this book is for these particular customers:

- Those who operate, design, or maintain a network that carries block, file, or object storage traffic
- Those who have experienced congestion in storage networks and are trying to educate themselves on this subject
- Those who want to learn the data-plane details of Fibre Channel, lossless Ethernet, and TCP
- Those who have a storage background but not much experience with TCP/IP networks
- Those who have TCP/IP background but not much experience in handling storage traffic
- Those who want to learn how different types of transports and networks handle storage traffic
- Those who are thinking about NVMe over Fabrics and are curious about its implications on network congestion

What This Book Is Not For

The focus of this book is on a particular topic—congestion—within a specific technology segment—storage networks. It is not a general book on storage networks. It does not explain how to design, configure, and operate a storage network. Those are detailed topics and probably require dedicated books.

In addition, this book is not intended to help you make a buying decision. In other words, we do not want to make it a protocol war. Fibre Channel, lossless Ethernet (FCoE, RoCE, and RoCEv2), and TCP have their use cases and serve their purposes when used correctly.

Keep in mind that this book is less focused on the control plane than on the data plane. It does not explain routing protocols, discovery mechanisms, security policies, Fibre Channel zoning, and so on. In addition, FICON and InfiniBand are beyond the scope of this book.

Finally, internal architecture and congestion within end devices, such as servers, host operating systems, storage arrays, and NAS devices, is beyond the scope of this book.

Prerequisites for This Book

If you are reading this book, you have probably experienced congestion in storage networks or one of its variants, such as slow drain, overutilization, or PFC storms.

We can't say this is a beginner's book. A basic understanding of storage architecture and its networks will be helpful. Those who have a limited background in these technologies can still benefit from this book without worrying about how these networks are configured. For example, this book does not explain configuration of zoning in Fibre Channel fabrics. Likewise, it does not explain configuration of quality of service in IP/Ethernet networks for transporting storage traffic.

The Case Studies in This Book

This book provides a number of case studies, and all of them are real. Over the years, we have worked with thousands of organizations to detect, troubleshoot, and prevent congestion in their production networks. We feature just a few selected case studies that we believe can help the entire community.

Focus on Block-Storage Traffic in a Network

This book focuses on block-storage traffic in a network for two reasons. First, block storage has the most stringent requirements among all types of storage traffic. If a network meets the requirements of block storage, it can very well exceed the requirements for file and object storage. Second, all types of storage traffic result in similar traffic patterns on a network. What you learn from block-storage networks for congestion management you can apply to other types of networks.

Fibre Channel Coverage

This is a book on Fibre Channel as much as it is on Ethernet and TCP. It actually dedicates more pages to Fibre Channel chapters than to Ethernet and TCP chapters—for a couple of reasons:

- Fibre Channel networks continue to be the most common networks for carrying block-storage traffic.
- Even if you do not use Fibre Channel, there is a lot to learn from it because Fibre Channel is used by all types of organizations around the world for transporting block-storage traffic. In addition, Fibre Channel has the longest history of transporting storage traffic among all the types of networks. It would be smart to learn from it and carry forward the same best practices.

Do not judge the lossless Ethernet and TCP chapters just by their page count. Many sections in those chapters refer to the earlier Fibre Channel chapters for details because the upper-layer protocols (SCSI and NVMe) are the same, regardless of the transport type. Their page counts would have been much higher had the earlier Fibre Channel chapters not already explained specific details.

Despite many claims and predictions, the reality is that Fibre Channel continues to be the most used network type for block-storage traffic in most data centers around the world.

Consider these facts: According to 2022 numbers, the Fibre Channel switching total addressable market (TAM) is worth approximately $2 billion annually. This TAM has not changed much in the past 15 years. In fact, every 4 to 5 years, the TAM increases by 5% to 8% due to speed upgrades (16 GFC to 32 GFC to 64 GFC). More importantly, Fibre Channel SANs account for only 10% to 15% of the overall external storage systems' expense, which was approximately $31 billion in 2022, and a vast majority of external storage devices connect to Fibre Channel SANs. There are investments also in servers and adapters that connect to the external storage arrays via Fibre Channel SANs. Besides having a stable market, Fibre Channel also has a future roadmap. As of this writing, the single-lane 128 GFC standard has been approved, and the 256 GFC standard is being developed.

We work with all kinds of organizations around the world that have hundreds of thousands of Fibre Channel ports deployed in their production environments. They use Fibre Channel SANs for critical Tier 1 workloads. We don't see these organizations moving away from Fibre Channel anytime soon—or even in the long term.

There are not many books on Fibre Channel. There are even fewer books explaining its practical use. These are the key reasons that many users lack a thorough understanding of congestion management. Hence, detecting, troubleshooting, and preventing congestion can be difficult for them. What is unknown is often perceived to be difficult.

Consider the following points:

- Fibre Channel and other variants of storage networks are rarely taught in colleges and universities. Hence, new industry talent does not get an opportunity to learn it.
- Basic books and courses on data communication start with three types of networks: LANs, WANs, and SANs. These days, almost everybody grows up seeing LANs around them, such as home and school Wi-Fi networks. They also see the Internet, which is a kind of WAN. However, people don't get an opportunity to work with SANs until they get in jobs that involve managing these environments.
- The so-called cloud wave has overshadowed other technologies, resulting in a narrative that storage networks and related technologies are irrelevant. Hence, new industry talent does not see a return on investment in learning it.

When new talent takes a job of managing storage infrastructure and networks, the learning options are limited. Existing books are dated. Theoretical explanations do not tend to focus on the practical details of managing production networks. Vendor documentation is geared toward product usage. Protocol specifications are difficult to read and aimed at product developers instead of users. For years, we have seen a demand from thousands of users for education on this topic. It just took us a while to execute our plan of writing this book.

How This Book Is Organized

Chapter 1, "Introduction to Congestion in Storage Networks," provides an overview of types of storage, storage protocols, their transports, and networks in a data center. It clarifies high-level concepts about NVMe over Fabric, quality of service (QoS), and congestion management in storage networks. This chapter also covers some questions that we have been asked over the years and our responses to them.

Chapter 2, "Understanding Congestion in Fibre Channel Fabrics," covers the following:

- Fibre Channel B2B flow control
- Sources of congestion, such as end devices, ISLs, and switches
- Causes of congestion, such as slow drain, overutilization, bit errors, and lack of credits on ISLs
- Effects of bit errors on congestion, details of data transmission on Fibre Channel fabrics, and data-plane counters for monitoring the health of links
- Forward Error Correction (FEC) and how it can provide insights for predicting congestion issues
- B2B credit loss recovery and B2B state change mechanisms

This chapter also provides a case study of an online retailer to illustrate the importance of proactive monitoring in storage networks.

Chapter 3, "Detecting Congestion in Fibre Channel Fabrics," covers the following:

- Congestion detection workflow and explains what, where, and how to detect congestion
- Congestion detection metrics such as TxWait, Slowport-monitor, and credit loss, with examples of Cisco MDS switches
- Automatic alerting and examples of the Port-Monitor feature on Cisco MDS switches
- Congestion detection on remote monitoring platforms, such as Cisco Nexus Dashboard Fabric Controller (NDFC) and custom-built apps like the MDS Traffic Monitor (MTM) app
- Metric export mechanisms for monitoring congestion
- Congestion detection on long-distance links

This chapter also discusses the pitfalls of monitoring network traffic and congestion.

Chapter 4, "Troubleshooting Congestion in Fibre Channel Fabrics," covers the following:

- Congestion severities, levels, and symptoms
- The types of victims, such as direct victims, indirect victims, and same-path victims

- Congestion detection methodology and a detailed workflow
- Hints and tips for troubleshooting congestion
- Cisco MDS NX-OS commands for troubleshooting congestion

This chapter demonstrates troubleshooting congestion in production networks with the help of multiple case studies.

Chapter 5, "Solving Congestion with Storage I/O Performance Monitoring," covers the following:

- The importance of storage I/O performance monitoring
- How and where to monitor storage I/O performance
- The basics of Cisco SAN Analytics
- I/O flows in Fibre Channel fabrics
- The basics of I/O flow metrics and some use cases
- SCSI and NVMe I/O operations and their effects on network traffic patterns and congestion

This chapter demonstrates the use of Cisco SAN Analytics in finding the root cause and predicting the likeliness of congestion by gaining I/O flow-level visibility into storage networks.

Chapter 6, "Preventing Congestion in Fibre Channel Fabrics," covers the following:

- Various approaches to eliminating or reducing congestion in storage networks
- Congestion recovery through disconnection of a culprit device
- Congestion recovery through early dropping of frames, using the congestion-drop timeout and no-credit-drop timeout features on Cisco MDS switches
- Congestion segregation through the use of techniques for segregating traffic to dedicated links or virtual links
- Automatic changing of traffic assignments for virtual links, using features such as the congestion isolation feature on Cisco MDS switches
- Congestion prevention using rate limiters on storage arrays
- Congestion prevention through the use of Dynamic Ingress Rate Limiting (DIRL) on Cisco MDS switches
- Congestion prevention through notification of end devices using Fibre Channel Fabric Performance Impact Notification (FPIN) frames and congestion signals
- Network design considerations, such as reducing the link speed of storage ports,

moving from edge–core–edge to collapsed-core designs, increasing traffic localization, and splitting large fabrics into smaller islands

In addition to providing a detailed explanation of various congestion prevention approaches, this chapter also demonstrates them in action and provides a case study of a bank preventing congestion in its storage networks.

Chapter 7, "Congestion Management in Ethernet Storage Networks," covers the following:

- Link-Level Flow Control (LLFC) and Priority Flow Control (PFC) in Layer 2 and Layer 3 networks, as well as the pause thresholds
- Ethernet flow control versus Fibre Channel flow control
- Congestion due to slow drain, overutilization of links, bit errors, and long-distance links in various network designs, such as a spine–leaf network
- Congestion detection metrics, such as the duration and the number of times traffic is paused, frame drops, bit errors, and link utilization
- I/O operations in FCoE and RoCE networks and their effects on network traffic and congestion
- Congestion troubleshooting in converged Ethernet networks with one or more no-drop traffic classes
- PFC storms
- Congestion prevention using pause timeout and PFC watchdog
- RoCEv2 Congestion Management (RCM)
- Congestion management when transporting lossless traffic in VXLAN

This chapter also explains the details of troubleshooting congestion in converged Ethernet networks when lossy and lossless traffic share the same network and the effect of one traffic type on the other.

Chapter 8, "Congestion Management in TCP Storage Networks," covers the following:

- Congestion in TCP storage networks with a spine–leaf network design
- I/O operations using iSCSI and NVMe/TCP and their effects on network traffic and congestion
- Congestion prevention in TCP storage networks, with an explanation of the practical use of Explicit Congestion Notification (ECN)
- Switch buffer management and active queue management mechanisms like Weighted Random Early Detection (WRED) and Approximate Fair Dropping (AFD)
- Congestion management with FCIP

This chapter focuses on block-storage traffic, especially for two types of users: those who have Fibre Channel experience but not much TCP/IP experience and those who have TCP/IP experience but not much experience handling storage traffic. This chapter provides a simplified explanation of TCP's reliable delivery, flow control, and congestion control and compares these concepts with Fibre Channel and lossless Ethernet networks. This chapter also provides an overview of nonstandard TCP implementations, such as DCTCP.

Chapter 9, "Congestion Management in Cisco UCS Servers," covers the following:

- Cisco UCS architecture, traffic flow, and flow control
- Congestion in a UCS domain
- The UCS Traffic Monitoring (UTM) app and its use in detecting and troubleshooting congestion in UCS servers

Even for those who do not use Cisco UCS, this chapter presents an excellent learning opportunity about congestion management in converged networks that carry lossless and lossy traffic on shared links. It discusses how congestion can be detected with minimal information using techniques like time-based trending and comparative analysis, and it provides case studies.

Credit

Figure 5.1: Dell Inc

This page intentionally left blank

Chapter 5

Solving Congestion with Storage I/O Performance Monitoring

This chapter explains the use of storage I/O performance monitoring for handling network congestion problems.

This chapter covers the following topics:

- Why Monitor Storage I/O Performance?
- How and Where to Monitor Storage I/O Performance.
- Cisco SAN Analytics Architecture
- Understanding I/O Flows in a Storage Network
- I/O Flow Metrics
- I/O Operations and Network Traffic Patterns
- Case studies

Why Monitor Storage I/O Performance?

Storage I/O performance monitoring provides advanced insights into network traffic, which can then be used to accurately address network congestion. This information is in addition to what the network ports already provide by counting the number of packets sent and received, the number of bytes sent and received, and link errors. In addition, storage I/O performance monitoring brings visibility to the upper layers of the stack and can explain why a network has or lacks traffic by providing the following information:

- The upper-layer protocol—SCSI or NVMe—that generated the network traffic
- Upper-layer protocol errors such as SCSI queue full, reservation conflict, NVMe namespace not ready, and so on

- IOPS, throughput, I/O size, and so on
- How long I/O operations take to complete, the delay caused by storage arrays, and the delay caused by hosts

This performance can also be monitored for every flow, giving granular insights into the traffic on a network port. This flow-level performance monitoring is extremely useful because most production environments are virtualized. When a host causes congestion due to overutilization of its link, the network can detect this condition, as explained in earlier chapters. In addition, storage I/O performance monitoring can detect the cause of the high amount of traffic and which virtual machine (VM) is asking for it.

Likewise, when a host causes congestion due to slow drain, investigating the SCSI- and NVMe-level performance and error metrics can explain why the host has become slower in processing the traffic. It is also possible to determine whether a particular VM has caused the entire host to slow down. In addition, storage I/O performance monitoring can also predict the likeliness of network congestion. These and many more benefits of storage I/O performance monitoring are explained in this chapter, and case studies are provided.

Storage I/O performance monitoring is a detailed subject. Its use cases involve application and storage performance insights, storage provisioning recommendations, infrastructure optimization, change management, audits, reporting, and so on. The scope of this book, however, is limited only to congestion use cases. We recommend continuing your education on this topic beyond this book. Refer to the References section later in this chapter.

This chapter focuses on the SCSI and NVMe protocols in the block-storage stack for performance monitoring. But these protocols initiate I/O operations only when an application wants them to read or write data. Therefore, monitoring higher layers in the stack, up to the application layer, can provide even more insights into why the network has traffic. Application-level monitoring, however—such as that provided by the Cisco AppDynamics observability platform—is beyond the scope of this book. This is another area that we recommend to continue your education outside this book.

How and Where to Monitor Storage I/O Performance

At a high level, storage I/O performance can be monitored within a host, in storage arrays, or in a network. These are three viable options because an I/O operation passes through many layers within the initiator (host), the target (storage array), and multiple switches in the network. This section explains these approaches briefly, but the primary focus of this chapter is on monitoring storage I/O performance in the network.

Storage I/O Performance Monitoring in the Host

Most operating systems, such as Linux, Windows, and ESXi, monitor storage I/O performance. Example 5-1 shows an example of monitoring storage I/O performance in Linux by using the **iotop** command.

Example 5-1 *Storage I/O Performance Monitoring in Linux*

```
[root@stg-tme-lnx-b200-7 ~]# iotop

Total DISK READ :      36.30 M/s | Total DISK WRITE :      36.85 M/s
Actual DISK READ:      36.31 M/s | Actual DISK WRITE:      36.80 M/s
  TID  PRIO  USER     DISK READ  DISK WRITE  SWAPIN     IO>    COMMAND
  941 be/3 root        0.00 B/s    0.00 B/s  0.00 %  3.31 % [jbd2/dm-101-8]
46303 be/4 root        6.42 M/s    6.37 M/s  0.00 %  1.93 % fio config_fio_1
  542 be/3 root        0.00 B/s    0.00 B/s  0.00 %  1.89 % [jbd2/dm-22-8]
26496 rt/4 root        0.00 B/s    0.00 B/s  0.00 %  1.26 % multipathd
46383 be/4 root        7.13 M/s    7.11 M/s  0.00 %  0.42 % fio config_fio_1
46284 be/4 root       11.96 M/s   12.34 M/s  0.00 %  0.00 % fio config_fio_1
46384 be/4 root        5.19 M/s    5.40 M/s  0.00 %  0.00 % fio config_fio_1
46402 be/4 root        5.61 M/s    5.63 M/s  0.00 %  0.00 % fio config_fio_1
```

For the purpose of dealing with network congestion, monitoring storage I/O performance within hosts involves the following considerations:

- Per-path storage I/O performance should be monitored because although multiple paths that perform at different levels exist between the host and the storage array, the host may, by default, report only cumulative performance.
- Metrics from thousands of hosts should be collected and presented in a single dashboard for early detection of congestion.
- Collecting the metrics from hosts may require dedicated agents, and there is overhead involved in maintaining them.
- Different implementations on different operating systems, such as Linux, Windows, and ESXi, may take non-uniform approaches to collecting the same metrics.
- Be aware that measuring the performance within hosts makes the measurements prone to issues on a particular host. Is the “monitored” end device “monitoring” itself? What happens when it gets congested or becomes a slow-drain device?
- Because of organizational silos, hosts and storage arrays may be managed by different teams.

Storage I/O Performance Monitoring in a Storage Array

Most arrays monitor storage I/O performance. For example, Figure 5-1 shows I/O performance on a Dell EMC PowerMax storage array.

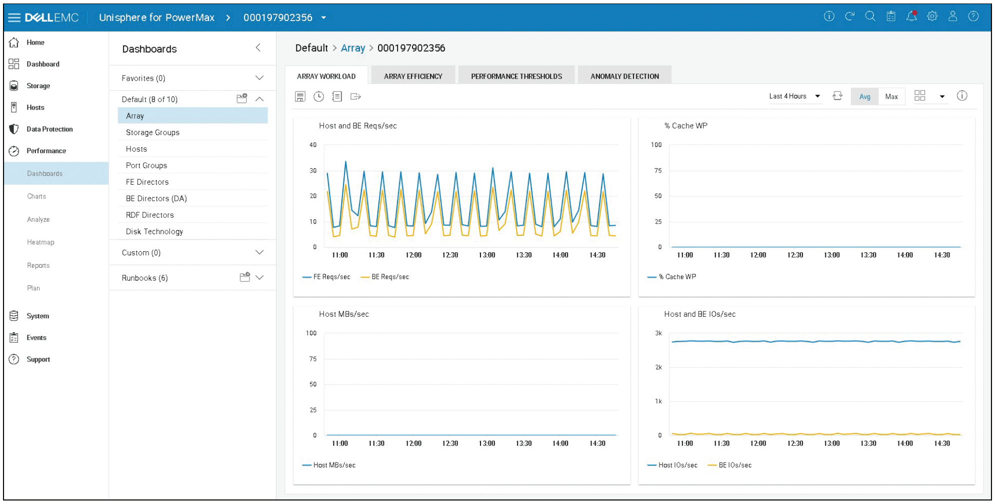

Figure 5-1 *Storage I/O Performance Monitoring on a Dell EMC PowerMax Storage Array*

The metrics collected by the storage arrays can be used for monitoring I/O performance, but this approach involves similar challenges to the host-centric approach, as explained in the previous section.

Storage I/O Performance Monitoring in a Network

I/O operations are encapsulated within frames for transporting the frames via a storage network. The network switches only need to look up the headers to send the frames toward their destination. In other words, a network, for its typical function of frame forwarding, need not know what's inside the frame. However, monitoring storage I/O performance in the network requires advanced capability on the switches for inspecting the transport (such as Fibre Channel) header, and upper-layer protocol (such as SCSI and NVMe) headers.

Cisco SAN Analytics monitors storage I/O performance natively within a network because it is integrated by design with Cisco MDS switches. As Fibre Channel frames are switched between the ports of an MDS switch, the ASICs (application-specific integrated circuits) inspect the FC and NVMe/SCSI headers and analyze them to collect I/O performance metrics such as the number of I/O operations per second, how long the I/O operations are taking to complete, how long the I/O operations are spending in the storage array, how long the I/O operations are spending in the hosts, and so on. Cisco SAN Analytics does not inspect the frame payload because there is no need for it, as the metrics can be calculated by inspecting only the headers.

Cisco SAN Analytics, because of its network-centric approach and unique architecture, has the following merits for monitoring storage I/O performance:

- **Vendor neutral:** Cisco SAN Analytics is not dependent on server vendor (HPE, Cisco, Dell, and so on), host OS vendor (Red Hat, Microsoft, VMware, and so on), or storage array vendor (Dell EMC, HPE, IBM, Hitachi, Pure, NetApp, and so on).
- **Not dependent on end-device type:** Cisco SAN Analytics is not dependent on any of the following:
  - **Server architecture:** Rack-mount, blade, and so on
  - **OS type:** Linux, Windows, or ESXi
  - **Storage architecture:** All-flash, hybrid, non-flash, and so on

  Legacy end devices can also benefit because no changes are needed on them, such as installation of an agent or firmware updates.
- **No dependency on the monitoring architecture of end devices:** Different products use different logic for collecting similar metrics. For example, some storage arrays collect I/O completion time on the front-end ports, whereas other storage arrays collect it on the back-end ports. Different host operating systems may collect I/O completion time at different layers in the host stack. Cisco SAN Analytics doesn't have this dependency.
- **Flow-level monitoring:** Cisco SAN Analytics monitors performance for every flow separately. When a culprit switchport is detected, flow-level metrics help in pinpointing the issue to an exact initiator, target, virtual machine, or LUN/namespace ID.
- **Flexibility of location of monitoring:** Cisco SAN Analytics can monitor storage I/O performance at any of the following locations:
  - **Host-connected switchports:** Close to apps and servers
  - **Storage-connected switchports:** Close to storage arrays
  - **ISL ports:** Flow-level granularity in the core of the network
- **Granular:** Cisco SAN Analytics monitors storage I/O performance at a low granularity—microseconds for on-switch monitoring and seconds for exporting metrics from the switch.

This chapter focuses on using Cisco SAN Analytics for addressing congestion in storage networks, although the education and case studies can be used with host-centric and storage array-centric approaches as well.

Cisco SAN Analytics Architecture

Cisco SAN Analytics architecture can be divided into three components (see Figure 5-2):

- Traffic inspection by ASICs on Cisco MDS switches
- Metric calculation by an onboard network processing unit (NPU) or by the ASIC
- Streaming of flow metrics to an external analytics and visualization engine for end-to-end visibility

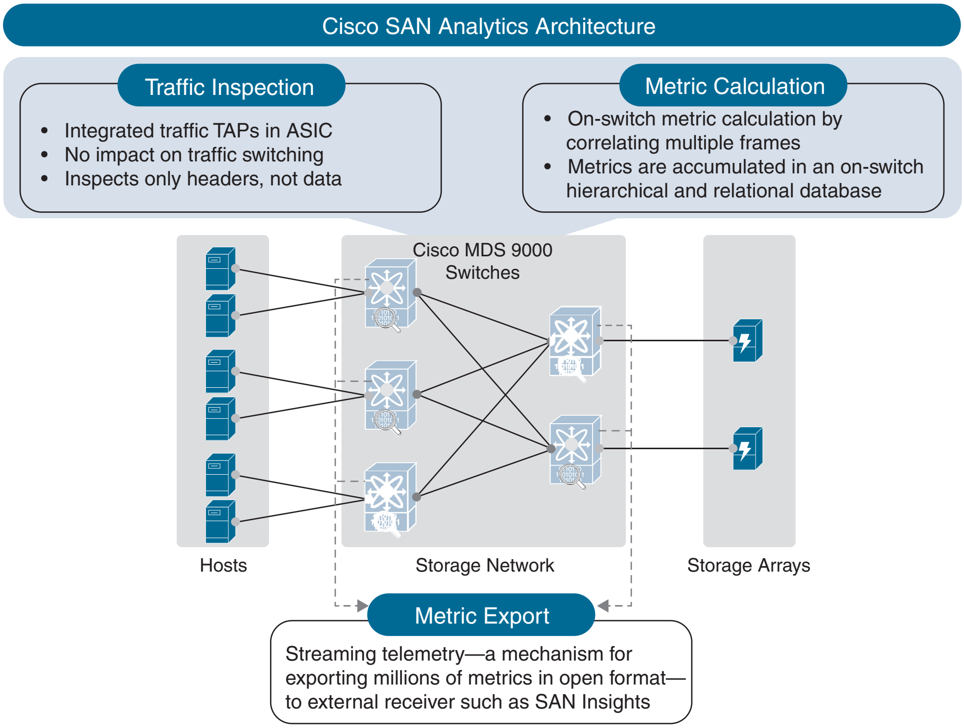

Figure 5-2 *Cisco SAN Analytics Architecture*

Traffic Inspection

Traffic inspection is integrated by design into Fibre Channel ASICs. In addition to switching the frames between the switchports, these ASICs can inspect the traffic in ingress and egress directions without any performance or feature penalty. In other words, traffic access points (TAPs) are built into the ASICs.

This approach is secure because the ASICs inspect only the Fibre Channel and SCSI/NVMe headers of the relevant frames. The frame payload (application data) is not inspected.

These ASICs are custom designed by Cisco, and they are exclusively used in MDS switches. Cisco Nexus switches and UCS fabric interconnects, despite supporting FC ports on selective models, use a different ASIC and thus don't offer SAN Analytics.

Metric Calculation

After inspecting the frame headers, Cisco MDS switches calculate the metrics by correlating multiple frames with common attributes, such as frames belonging to the same I/O operation and frames belonging to the same flow.

The metric calculation logic in the 32 Gbps MDS switches resides in an onboard network processing unit (NPU), which is a powerful packet processor. In 64 Gbps MDS switches, the metric calculation logic resides within the ASIC itself, although the NPU continues to exist on the switches. Regardless of this architectural detail, the overall metric calculation logic remains the same.

Cisco MDS switches accumulate the metrics in a hierarchical and relational database for on-switch visibility or export to a remote receiver.

Note At the time of this writing, Cisco SAN Analytics does not collect I/O flow metrics in FICON environments.

Metric Export

Cisco SAN Analytics is designed to inspect every flow that passes through a storage network in an always-on fashion. As a result, it collects millions of metrics per second. A traditional approach (such as SNMP) for exporting a large number of metrics may not work at this scale, and thus, Cisco introduced streaming telemetry for this purpose. In addition to being efficient, streaming telemetry exports metrics in open format, which simplifies third-party integrations.

The receiver of streaming telemetry can use I/O flow metrics from multiple switches to provide fabric-wide and end-to-end visibility into a single pane of glass for long-term metric retention, trending, correlation, predictions, and so on. SAN Insights is an example of such a receiver and is a feature in Cisco Nexus Dashboard Fabric Controller (NDFC), formerly known as Cisco Data Center Network Manager (DCNM). Figure 5-3 shows the SAN Insights dashboard, which provides many ready-made use cases, such as automatic learning, baselining, and deviation calculations for up to 1 million I/O flows per NDFC server as of release 12.1.2. This high scale gives visibility into issues anywhere in the fabric.

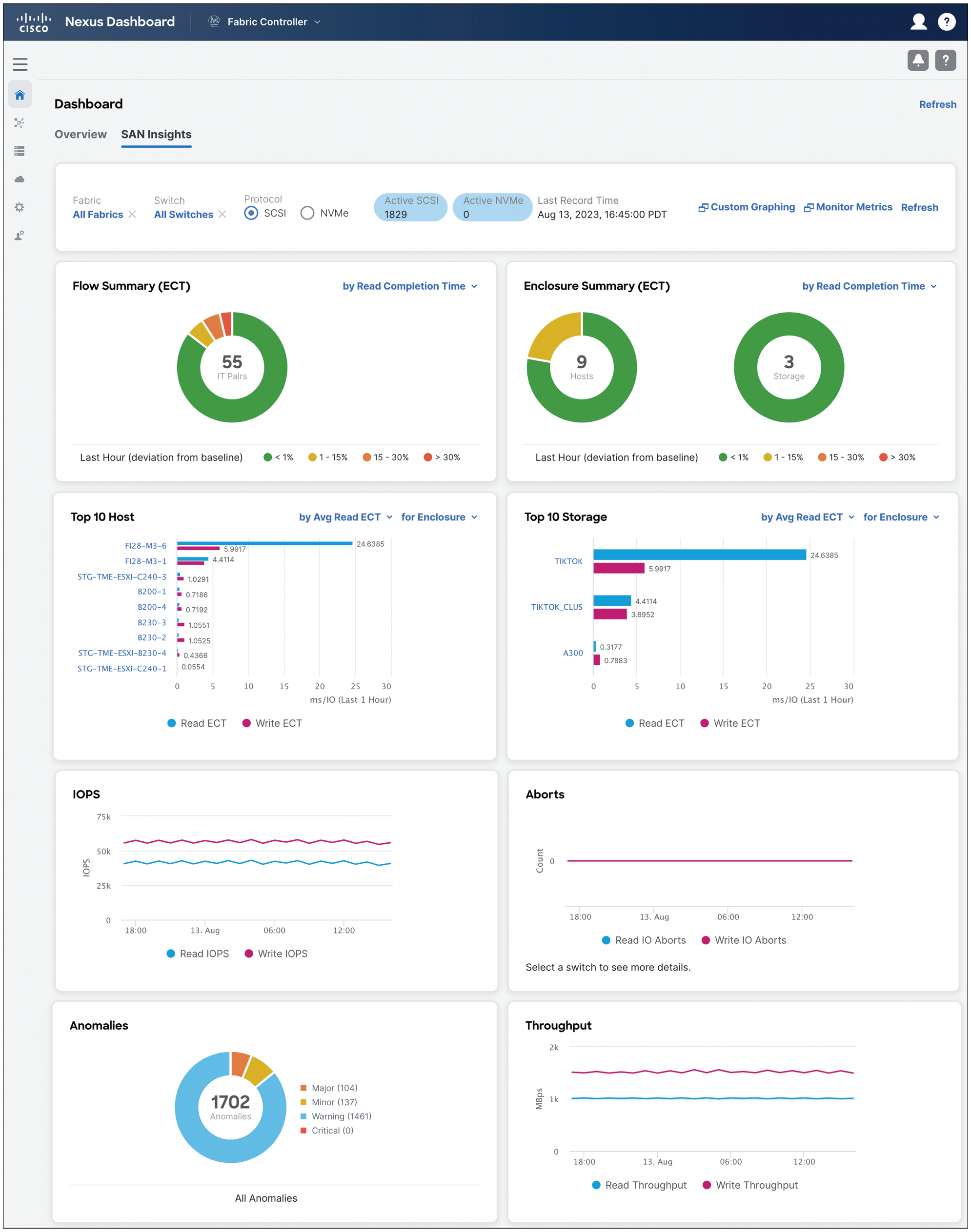

Figure 5-3 *SAN Insights Dashboard in Cisco NDFC*

Understanding I/O Flows in a Storage Network

Without considering I/O flows, a network is only aware of the frames in ingress and egress directions. Categorizing network traffic into I/O flows helps in correlating it with initiators, targets, and the logical unit number (LUN) for SCSI I/O operations and namespace ID (NSID) for NVMe I/O operations. In addition, storage performance can be monitored for every I/O flow individually to get detailed insights into the traffic. For example, when a switchport is 90% utilized, throughput per I/O flow can tell which initiator, target, and LUN/namespace are the top consumers.

I/O Flows in Fibre Channel Fabrics

The following can be the I/O flow types in a Fibre Channel fabric:

- **Port flow:** Traffic belonging to all the I/O operations that pass through a network port makes a port flow. It can an SCSI port flow for SCSI traffic or an NVMe port flow for NVMe traffic.
- **VSAN flow:** A port of a Cisco Fibre Channel switch may carry traffic in one or more VSANs. Hence, a port flow can be further categorized into one or more VSAN flows.
- **Initiator flow:** Traffic belonging to all the I/O operations that are initiated by an initiator makes an initiator flow.
- **Target flow:** Traffic belonging to all the I/O operations that are destined for a target makes a target flow.
- **Initiator-target (IT) flow:** Traffic belonging to all the I/O operations between a pair of initiator and target makes an IT flow.
- **Initiator-target-LUN (ITL) flow:** Traffic belonging to all the I/O operations between an initiator, a target, and a logical unit makes an ITL flow. An ITL flow is applicable only for SCSI I/O operations.
- **Initiator-target-namespace (ITN) flow:** Traffic belonging to all the I/O operations between an initiator, a target, and a namespace makes an ITN flow. An ITN flow is applicable only for NVMe I/O operations.
- **Target-LUN (TL) flow:** Traffic belonging to all the I/O operations that are destined for a target port and a specific logical unit makes a TL flow. A TL flow is applicable only for SCSI I/O operations.
- **Target-namespace (TN) flow:** Traffic belonging to all the I/O operations that are destined to a target port and a specific namespace makes a TN flow. A TN flow is applicable only for NVMe I/O operations.

The definition of an I/O flow can also be extended to a virtual entity (VE), such as a virtual machine (VM) on the host. When combined with an ITL or ITN flow, the end-to-end flow becomes a VM-ITL flow or a VM-ITN flow. There are at least two approaches for achieving this visibility into the VMs.

The first approach needs support from hosts, and in some cases even from storage arrays, for tagging the VM identifier in the frame header. Although Cisco SAN Analytics on MDS switches supports VM-ITL and VM-ITN flows, because of the dependency on the end devices, most production deployments are not ready for it at the time of this writing.

The second approach uses the APIs from VMware vCenter to provide the correlation between the VM and the initiator and LUN (or namespace) from the ITL (or ITN) flow. The benefit of this approach, unlike the first approach, is that upgrading the end devices is not mandatory. Cisco SAN Insights uses this approach in NDFC 12.1.2 onward.

In environments where even the read-only access to VMware vCenter cannot be added to NDFC, this approach can still be used for manually correlating ITL or ITN flows with the VMs. The use of this approach is demonstrated further in the section “Case Study 3: An Energy Company That Eliminated Congestion Issues,” later in this chapter.

This chapter focuses only on ITL flows that are natively available on the Cisco MDS switches without any dependency on the end devices and NDFC. The environments with VM-ITL flows made available using either of the two approaches mentioned earlier can benefit by expanding ITL flows in the same way that port flows are expanded to IT flows and ITL flows.

To understand the I/O flows and how they help in gaining granular details about a network, consider the example in Figure 5-4. Two initiators, I-1 and I-2, connect to two targets, T-1, and T-2, via a fabric of Switch-1 and Switch-2. The ISL port on Switch-1 (Port-3) reports an ingress throughput of 800 MBps. After enabling SAN Analytics, Port-3 can categorize network traffic into multiple types of I/O flows and monitor the performance of every flow.

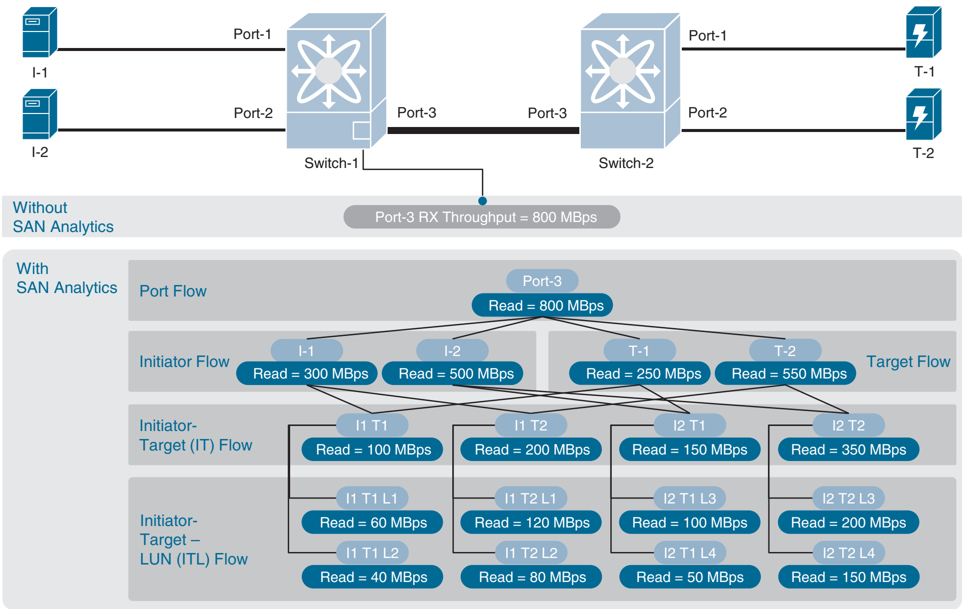

Figure 5-4 *I/O Flows and Flow-Level Metrics Using Cisco SAN Analytics*

SAN Analytics can find the following details:

- The 800 MBps throughput on Port-3 on Switch-1 is because of SCSI read I/O operations.
- Port-3 may have two VSANs: VSAN 100 and VSAN 200 (not shown in Figure 5-4). The VSAN flows provide a further breakdown of the port flow throughput, such as a read throughput of 600 MBps for VSAN 100 and a read throughput of 200 MBps for VSAN 200.
- I-1's read throughput via Port-3 is 300 MBps, whereas I-2's read throughput via Port-3 is 500 MBps.
- T-1's read throughput via Port-3 is 250 MBps, whereas T-2's read throughput via Port-3 is 550 MBps.
- Port-3 has four IT flows: I1-T1, I1-T2, I2-T1, and I2-T2. The read throughput for each is as follows:
  - **I1-T1:** 100 MBps
  - **I1-T2:** 200 MBps
  - **I2-T1:** 150 MBps
  - **I2-T2:** 350 MBps
- Port-3 has eight ITL flows. I-1 uses LUN-1 and LUN-2, whereas I-2 uses LUN-3 and LUN-4. The read throughput for each is as follows:
  - **I1-T1-L1:** 60 MBps
  - **I1-T1-L2:** 40 MBps
  - **I1-T2-L1:** 120 MBps
  - **I1-T2-L2:** 80 MBps
  - **I2-T1-L3:** 100 MBps
  - **I2-T1-L4:** 50 MBps
  - **I2-T2-L3:** 200 MBps
  - **I2-T2-L4:** 150 MBps

As is evident from this example, the hierarchical and relational definitions of I/O flows help create a precise breakdown of traffic on a switchport. During congestion, the per-flow metrics, such as throughput, help in pinpointing the root cause of the exact entity, such as initiator, target, LUN, or namespace. Without per-flow storage I/O performance monitoring, as provided by Cisco SAN Analytics, such detailed insights are not possible.

I/O Flows Versus I/O Operations

I/O flows shouldn't be confused with I/O operations. An I/O flow is identified by end-to-end tuples such as initiator, target, LUN, or namespace (ITL or ITN flows). In contrast, I/O operations transfer data within an I/O flow. For example, when Initiator-1 initiates 100 read I/O operations per second to LUN-1 on Target-1, the ITL flow is identified as Initiator-1–Target-1–LUN-1, whereas there were 100 I/O operations per second.

An I/O flow is created only after an initial exchange of I/O operations between the identifying tuples. Later, if the initiator doesn't read or write data, the I/O flows may still exist, but no I/O operations flow through it, which results in zero IOPS for these I/O flows.

I/O Flow Metrics

The I/O flow metrics collected by Cisco SAN Analytics can be classified into the following categories:

- **Flow identity metrics:** These metrics identify a flow, such as switchport, initiator, target, LUN, or namespace.
- **Metadata metrics:** The metadata metrics provide additional insights into the traffic. For example:
  - **VSAN count:** Number of VSANs carrying traffic on a switchport.
  - **Initiator count:** Number of initiators exchanging I/O operations behind a switchport.
  - **Target count:** Number of targets exchanging I/O operations behind a switchport.
  - **IT flow count:** Number of pairs of initiators and targets exchanging I/O operations via a switchport.
  - **TL and TN flow count:** Number of pairs of targets and LUNs/namespaces behind a switchport exchanging I/O operations.
  - **ITL and ITN flow count:** Number of pairs of initiators, targets, and LUNs/namespaces exchanging I/O operations via a switchport.
  - **Metric collection time:** Start time and the end time for I/O flow metrics during a specific export. This metric helps in knowing the precise duration when a metric was calculated at the link.
- **Latency metrics:** Latency metrics identify the total time taken to complete an I/O operation and the time taken to complete various steps of an I/O operation. For example:
  - **Exchange Completion Time (ECT):** Total time taken to complete an I/O operation.
  - **Data Access Latency (DAL):** Time taken by a target to send the first response to an I/O operation. DAL is one component of ECT that's caused by the target.

- **Host Response Latency (HRL):** Time taken by an initiator to send the response after learning that the target is ready to receive data for a write I/O operation. HRL is one component of ECT that's caused by the initiator.

- **Performance metrics:** These metrics measure the performance of I/O operations. For example:
  - **IOPS:** Number of read and write I/O operations completed per second.
  - **Throughput:** Amount of data transferred by read and write operations, in bytes per second.
  - **Outstanding I/O:** The number of read and write I/O operations that were initiated but are yet to be completed.
  - **I/O size:** The amount of data requested by a read or write I/O operation.
- **Error metrics:** The error metrics indicate errors in read and write I/O operations (for example, Aborts, Failures, Check condition, Busy condition, Reservation Conflict, Queue Full, LBA out of range, Not ready, and Capacity exceeded).

An exhaustive explanation of all these metrics is beyond the scope of this chapter. This chapter is just a starting point for using end-to-end I/O flow metrics in solving congestion and other storage performance issues.

Latency Metrics

Latency is a generic term to convey storage performance. But as Figure 5-5 and Figure 5-6 show, there are multiple latency metrics, each conveying a specific meaning. Latency metrics are measured in time (microseconds, milliseconds, and so on).

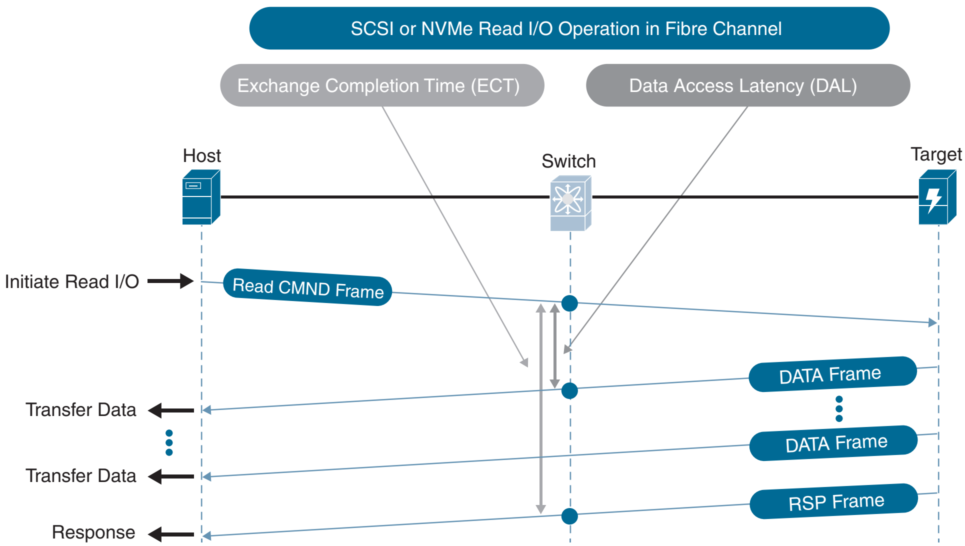

Figure 5-5 *Latency Metrics for a Read I/O Operation*

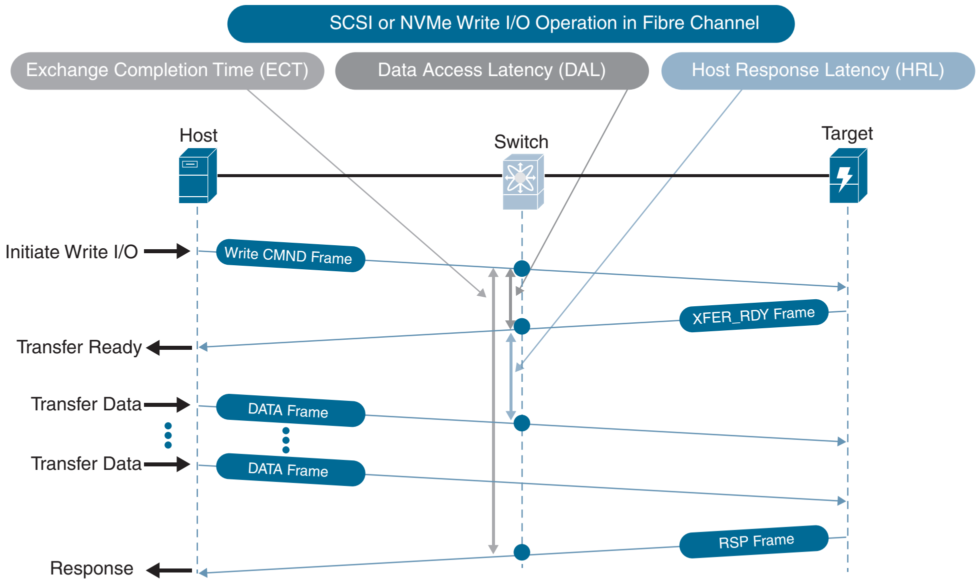

Figure 5-6 *Latency Metrics for a Write I/O Operation*

Exchange Completion Time

Exchange Completion Time (ECT) is the time taken to complete an I/O operation. It is a measure of the time difference between the command (CMND) frame and the response (RSP) frame. In Fibre Channel, an I/O operation is carried out by an exchange, and hence it's called Exchange Completion Time, but ECT can also be known as I/O completion time.

ECT is an overall measure of storage performance. In general, the lower the ECT, the better. This is because lower ECTs result in improved application performance.

At the same time, a direct correlation between ECT and application performance is not straightforward because it's dependent on the application I/O profile. In general, when application performance degrades and if ECT increases (degrades) at the same time, the reason for the performance degradation is the slower I/O performance.

Data Access Latency

Data Access Latency (DAL) is the time taken by a storage array in sending the first response after receiving a command (CMND) frame. For a read I/O operation, DAL is calculated as the time difference between the command (CMND) frame and the first-data (DATA) frame. For a write I/O operation, DAL is calculated as the time difference between the command (CMND) frame and the transfer-ready (XFER_RDY) frame.

When a target receives a read I/O operation, if the data requested is not in cache, the target must first read the data from the storage media, which takes time. The amount of time it takes to retrieve the data from the media depends on several factors, such as overall system utilization and the type of storage media being used. Likewise, when a target receives a write I/O operation, it must process all the other operations ahead of this operation, which takes time. An increase in these time values leads to a large DAL.

In most cases, it's best to investigate DAL while troubleshooting higher ECT because DAL may tell why ECT increased. An increase in ECT and also in DAL indicates a slowdown within the storage array.

Host Response Latency

Host Response Latency (HRL), for a write I/O operation, is the time taken by a host in sending the data after receiving the transfer ready. It is calculated as the time difference between the transfer-ready frame and the first data frame.

Because read I/O operations do not have transfer ready, HRL is not calculated for them.

In most cases, it's best to investigate HRL while troubleshooting higher-write ECTs because HRL may tell why ECT increased. An increase in write ECT and also in HRL indicates a slowdown within the host.

Using Latency Metrics

The following are important details to remember about latency metrics, such as ECT, DAL, and HRL, when addressing congestion in a storage network:

- A good way of using ECT is to monitor it for a long duration and find any deviations from the baseline. For example, consider two applications with an average ECT of 200 µs and 400 µs over a week. The I/O flow path of the first application gets congested, resulting in an increased ECT of 400 µs. At this moment, although both applications have the same ECT, only the first application may be degraded, while the second application remains unaffected, even though their ECT values are the same.
- ECT measures the overall storage performance, but it doesn't convey the source of the delay, which can be the host, network, or storage array. The delay caused by the host is measured by HRL, whereas the delay caused by the storage array is measured by DAL.
- The delay caused by the network may be the direct result of congestion. For example, when a host-connected switchport has high TxWait, the frames can't be delivered to it in a timely fashion. As a result, the time taken to complete the I/O operations (ECT) increases.
- Although an increase in TxWait (or a similar network congestion metric) increases ECT, the reverse may not be correct. ECT may increase even when the network isn't congested. ECT is an end-to-end metric. It may increase due to delays caused by hosts, network, or storage. The block I/O stack within a host involves multiple layers. Similarly, an I/O operation undergoes many steps within a storage array. The delay caused by any of these layers increases ECT.

- Network congestion is one of the reasons for higher ECT. However, it's not the only reason. Other network issues may increase ECT even without congestion (for example, network traffic flowing through suboptimal paths, long-distance links, or poorly designed networks).
- All latency metrics increase under network congestion. This increase is seen in all the I/O flows whose paths are affected by congestion.
- While considering dual fabrics with active/active multipath, if only one fabric is congested, only the I/Os using the congested fabric report increases in ECT. The average increase in the ECT as reported by the host may or may not show this difference, depending on how much ECT degrades. For example, consider an application that measures I/O completion time (ECT) as 200 µs. The application accesses storage via Fabric-A and Fabric-B. ECT over Fabric-A is 180 µs, whereas ECT over Fabric-B is 220 µs. If Fabric-A becomes congested, resulting in an increase in ECT from 180 to 270 µs (50% deviation), the average ECT as measured by the application increases to 245 µs, which is only a 22% increase.

How can you verify if an increase in ECT for an application is because of congestion or not? Here are some suggestions:

- Check the metrics for the ports (such as TxWait) in the end-to-end data path.
- Check the ECT of the I/O flows that use the same network path as the switchport. If ECT increases just for one I/O flow but the rest of the I/O flows don't show an increase, it is not a network congestion issue because the network doesn't do any preferential treatment for I/O flows. A fabric just understands the frames, and all frames are equal for it.
- Investigate other metrics, like I/O size, IOPS, and so on. A common example is an increase in I/O size because larger I/O size operations take longer to complete. Also, find any SCSI and NVMe errors and link-level errors.

The Location for Measuring Latency Metrics

Cisco SAN Analytics calculates latency metrics by taking the time difference between relevant frames on the analytics-enabled switchports on MDS switches. As a result, the absolute value of these metrics may differ by a few microseconds, depending on the exact location of the measurement. For example, the ECT reported by a storage-connected switchport may be a few microseconds lower than the ECT reported by a host-connected switchport. This is because the storage-connected switchport sees the command frame a few microseconds after the host-connected switchport does, and it sees the response frames a few microseconds earlier than the host-connected switchport. When the time difference between the command frame and the response frame on the storage port is considered, it comes out to be less than the time difference between the command frame and the response frame on the host-connected switchport.

This difference in the value of latency metrics based on the location of measurement is marginal. It may be a matter of discussion in an academic exercise, but for any real-world production environment, the difference is very small, increases complexity, makes it hard for various teams to understand the low-level details, and doesn't change the end result.

What is more important is to understand that in lossless networks, congestion spreads from end to end quickly. If this congestion increases ECT by 50% on the storage-connected switchport, the same percentage increase will be seen on the host-connected port also, although the absolute values may differ.

What happens if the congestion is only severe enough that the effect is limited to storage ports or host ports? In production environments, the spread of congestion can't be predicted. More importantly, if the congestion has not spread from end to end, it's not severe enough to act on. In such cases, it is best to monitor and use the metrics for future planning, but without an end-to-end spread, the effect of congestion is limited to a small subset of the fabric.

Performance Metrics

Performance metrics convey the rate of I/O operations, their pattern, and the amount of data transferred.

I/O Operations per Second (IOPS)

IOPS, as its name suggests, is the number of read or write I/O operations per second. Typically, IOPS is a function of the application I/O profile and the type of storage. For example, transactional applications have higher IOPS requirements than do backup applications. Also, SSDs provide higher IOPS than do HDDs.

It is not possible to infer the network traffic directly from IOPS. An I/O operation may result in a few or many frames, depending on the data transferred by that I/O operation. Likewise, the throughput caused by I/O operations depends on the amount of data transferred by those I/O operations. Hence, it's difficult to predict the effect of higher IOPS on network congestion without accounting for I/O size, explained next.

On the other hand, network congestion typically results in reduced IOPS because the network is unable to deliver the frames to their destinations in a timely fashion or can transfer fewer frames.

I/O Size

The amount of data transferred by an I/O operation is known as its I/O size. I/O size is a function of the application's I/O profile. For example, a transactional application may have an I/O size of 4 KB, whereas a backup job may use an I/O size of 1 MB.

This I/O size metric in the context of storage I/O performance monitoring or SAN Analytics is different from the amount of data that an application wants to transfer as part of an application-level transaction or operation. For example, an application may want to transfer 1 MB of data, but the host may decide to request this data using four I/O operations, each of size 256 KB. This difference is worth understanding, especially while investigating various layers within a host.

I/O size is encoded in the command frame of I/O operations. It has no dependency on network health. As a result, I/O size doesn't change with or without congestion.

Large I/O size results in a higher number of frames, which in turn leads to higher network throughput. For example, a 2 KB read I/O operation results in just one Fibre Channel data frame of size 2 KB, whereas a 64 KB read I/O operation results in 32 Fibre Channel frames of size 2 KB. Because of this, I/O size directly affects the network link utilization and thus provides insights into why a host port or a host-connected switchport may be highly utilized. For example, a host link may not be highly utilized with an I/O size of 16 KB. But the same link may get highly utilized and thus become the source of congestion when the I/O size spikes to 1 MB.

To understand the effect of I/O size on link utilization, consider the example in Figure 5-7. Two hosts, Host-1, and Host-2, connect to the switchports at 8 GFC to access storage from multiple arrays. Both servers are doing 10,000 read I/O operations per second (IOPS). However, the I/O sizes used by the two servers are different. Host-1 uses an I/O size of 4 KB, whereas Host-2 uses an I/O size of 128 KB.

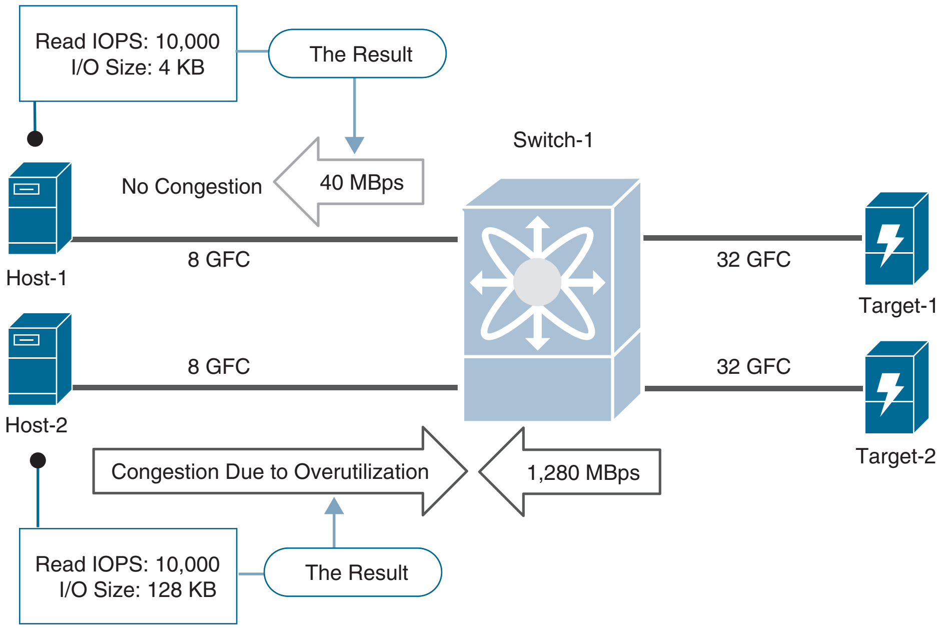

Figure 5-7 *Detecting and Predicting the Cause of Congestion Using I/O Size*

Host-1, with 10,000 IOPS and 4 KB I/O size, results in a throughput of 40 MBps, whereas Host-2, with 10,000 IOPS and 128 KB I/O size, results in a throughput of 1280 MBps. As evident, 1280 MBps can't be transported via an 8 GFC link because its maximum data rate is 800 MBps. As a result, Host-2's read I/O traffic causes congestion due to overutilization. Host-1 doesn't cause congestion even though its read IOPS is the same as Host-2's. I/O size is the differentiating factor here.

Throughput

Throughput is a generic term that has different meanings for different people. For measuring storage performance, throughput is measured as the amount of data transferred by I/O operations, in megabytes per second (MBps). On the other hand, for measuring network performance, throughput is measured in frames transferred per second and the amount of data transferred by those frames, in gigabits per second (Gbps).

Note Pay attention to measuring storage performance in bytes (B) per second and network performance in bits (b) per second and don't forget to convert from bytes to bits or vice versa.

Another important detail to remember is that the read and write I/O throughput may have a marginal difference when measured on the end devices versus on the network. Applications measure the total amount of data that they exchange with the storage volumes. However, the network throughput differs slightly because I/O operations have headers, such as Fibre Channel headers and SCSI/NVMe headers. For all practical purposes, this marginal difference can be ignored. Be aware that the throughput reported by various entities may differ but don't get carried away by these marginal differences.

Outstanding I/O

Outstanding I/O is the number of I/O operations that were initiated but are yet to be completed. In other words, an initiator sent a command frame, but it hasn't received a response frame yet. Outstanding I/O is also known as open I/O or active I/O.

In production environments, there are always new I/Os being originated while the previous I/Os are being completed because the applications may be multithreaded or multiprocessed. Also, keeping some I/O operations open helps in a performance boost.

Outstanding I/O is directly related to the queue-depth value on a host as well as similar values on storage arrays. Different entities have different thresholds for outstanding I/O. For example, a host may stop initiating new I/O operations when the outstanding I/O reaches a threshold, such as 32. Likewise, a target may reject new incoming I/O operations when a large number of I/O operations (such as 2048) are already open (or outstanding), and the target is still processing them.

Congestion in a storage network may be a side effect of a large number of outstanding I/O operations.

I/O Operations and Network Traffic Patterns

Traffic in a storage network is the direct result of an application initiating a read or write I/O operation. Because of this, network traffic patterns can be better understood by analyzing the application I/O profile, such as the timing, size, type, and rate of I/O operations. Essentially, the application I/O profile helps in understanding why the network has traffic.

Read I/O Operation in a Fibre Channel Fabric

Figure 5-8 shows a SCSI or NVMe read I/O operation in a Fibre Channel fabric. A host initiates a read I/O operation using a read command, which the host encapsulates in a Fibre Channel frame and sends out its port. The host-connected switchport receives the frame and sends them to the next hop, based on the destination in the frame header. The network of switches, in turn, delivers this frame to the target. Such a frame that carries a read command is called a read command frame (CMND).

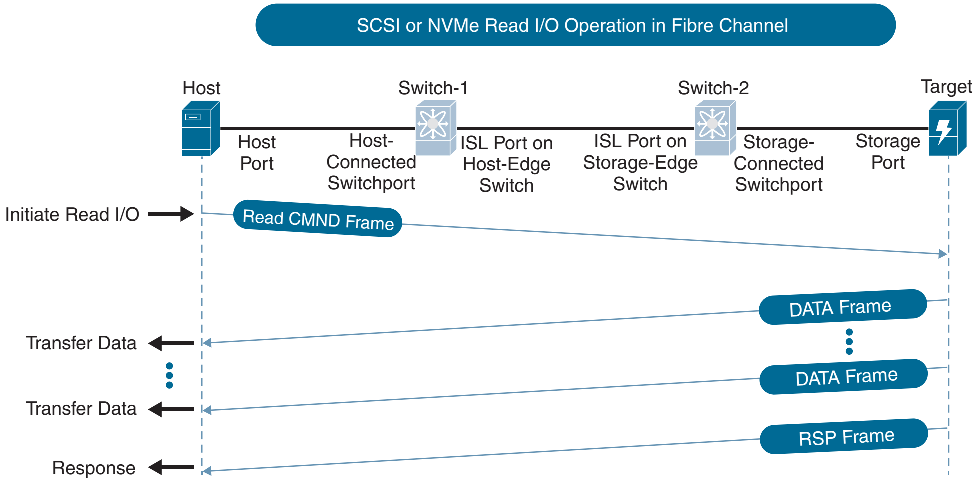

Figure 5-8 *SCSI or NVMe Read I/O Operation in a Fibre Channel Fabric*

The target, after receiving the read command frame, sends the data to the host in one or more FC frames. These frames that carry data are called data frames (DATA). The exact number of data frames returned by the target depends on the I/O size of the read command. A full-size FC frame can transfer up to 2048 bytes (2 KB) of data. Hence, the target sends one data frame if the read I/O size is less than or equal to 2 KB. The size of this frame depends on the data carried by it plus the overhead of the header. However, when the I/O size is larger than 2 KB, the target sends the data in multiple frames. Typically, all these frames are full-size FC frames carrying 2 KB worth of data. If the size requested is not a multiple of 2 KB, then the last frame is smaller than 2 KB. For example, an I/O size of 4 KB results in two full-size FC frames. But if the I/O size is 5 KB, the target may send two full-size FC frames, each carrying 2 KB, and a third frame carrying any remaining data, which is 1 KB.

After sending all the data to the host, the target indicates the completion of the I/O operations by sending a response, which carries the status. A frame that carries a response is called a response frame (RSP).

Some implementations can optimize the read I/O operations by sending the last data and the response in the same frame if their combined size is below 2 KB. These optimized read I/O operations may not always have dedicated response frames. Regardless of the type of read I/O operation, their result on network traffic remains the same.

Write I/O Operation in a Fibre Channel Fabric

Figure 5-9 shows a SCSI or NVMe write I/O operation in a Fibre Channel fabric. A host initiates a write I/O operation using a write command, which the host encapsulates in a Fibre Channel frame and sends out its port. The host-connected switchport receives the frame and sends it to the next hop, based on the destination in the frame header. The network of switches, in turn, delivers this frame to the target. Such a frame that carries a write command is called a write command frame (CMND).

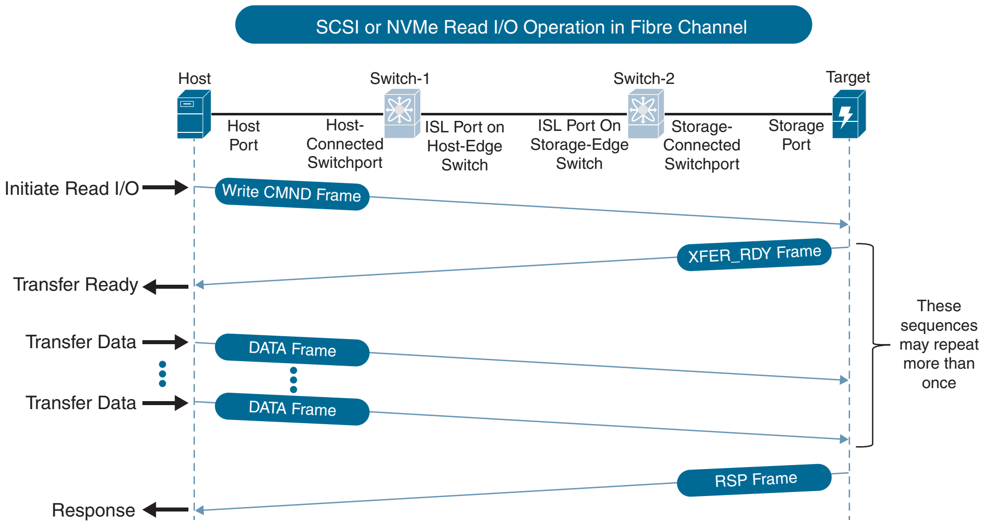

Figure 5-9 *SCSI or NVMe Write I/O Operation in a Fibre Channel Fabric*

The target, after receiving the write command frame, prepares to receive the data and sends a frame to the host indicating that it is ready to receive all or some of the write data. This is called a transfer-ready frame (XFER_RDY). A transfer-ready frame carries the amount of data that the target is ready to receive in one sequence or burst. Refer to Chapter 2, "Understanding Congestion in Fibre Channel Fabrics," for more details on a Fibre Channel sequence. Typically, this size is the same as the size requested by the write command frame. But sometimes, the target may not have the resources to receive all the data that the host wants to write in a single sequence. For example, a host may want to write 4 MB of data, which it specifies in the write command frame. The target, however, may have the resources to accept only 1 MB of data at a time. Hence, the target sends 1 MB as the burst length in the transfer-ready frame.

The host, after receiving the transfer-ready frame, sends the data to the host in one or more FC frames. These frames are called data frames (DATA). The exact number of data frames returned by the host depends on the burst size of the transfer-ready frame. It follows the same rules as explained previously for the read I/O operations. The difference for write I/O operations is that multiple sequences of transfer-ready may be involved if the target chooses to return a burst size that is less than the write command I/O size.

After receiving all the data that the host requested to write in this I/O operation (which may have been in multiple sequences due to the target sending one or multiple transfer-ready frames), the target indicates the completion of the I/O operations by sending a response, which carries the status. A frame that carries a response is called a response frame (RSP).

Some implementations can optimize the write I/O operations by eliminating the transfer-ready frame. In such cases, the target informs the initiator, during the process login (PRLI) state, that it will always keep the resources ready to receive a minimum size (first burst) of data. The initiator sends the data frames immediately after sending the write command frames, without waiting for the transfer-ready frames to arrive. Regardless of the type of write I/O operation, the result on network traffic is the same.

Network Traffic Direction

Table 5-1 shows the direction of traffic as a result of a read I/O operation in Figure 5-8. Figure 5-10 shows the traffic directions on various network ports due to different sequences of read and write I/O operations.

Table 5-1 *Traffic Direction in a Storage Network Because of Read I/O Operation*

Frame Type	Host Port	Host-Connected Switchport	ISL Port on Host-Edge Switch	ISL Port on Storage-Edge Switch	Storage-Connected Switchport	Storage Port
Read I/O command frame	Egress	Ingress	Egress	Ingress	Egress	Ingress
Read I/O data frame	Ingress	Egress	Ingress	Egress	Ingress	Egress
Read I/O response frame	Ingress	Egress	Ingress	Egress	Ingress	Egress

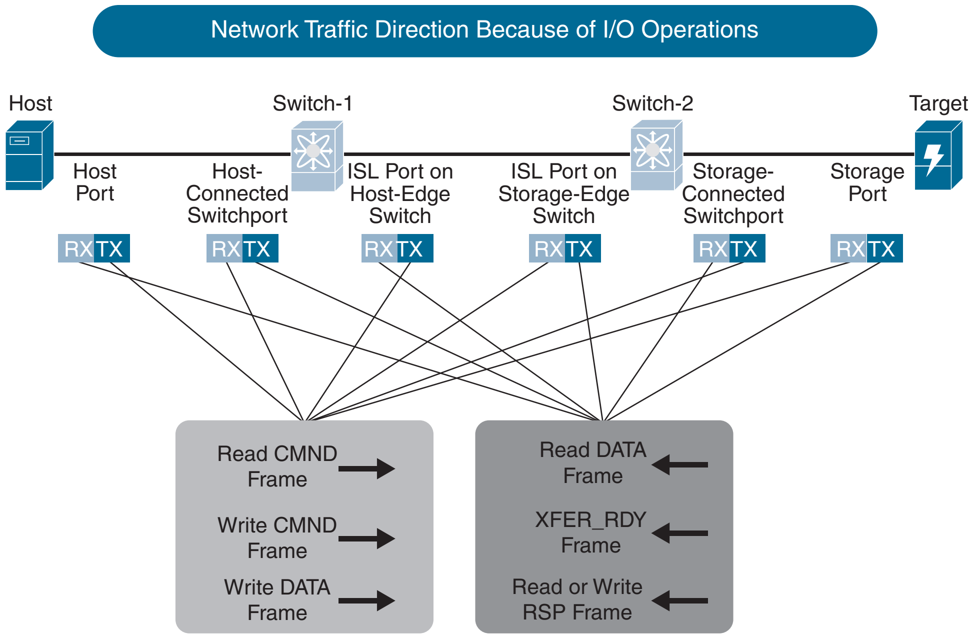

Figure 5-10 *Network Traffic Direction Because of Read and Write I/O Operations*

Table 5-2 explains the direction of traffic because of a write I/O operation in Figure 5-9. Figure 5-10 shows the traffic directions on various network ports due to different sequences of read and write I/O operations.

Table 5-2 *Traffic Direction in a Storage Network Because of Write I/O Operation*

Frame Type	Host Port	Host-Connected Switchport	ISL Port on Host-Edge Switch	ISL Port on Storage-Edge Switch	Storage-Connected Switchport	Storage Port
Write I/O command frame	Egress	Ingress	Egress	Ingress	Egress	Ingress
Write I/O transfer ready	Ingress	Egress	Ingress	Egress	Ingress	Egress
Write I/O data frame	Egress	Ingress	Egress	Ingress	Egress	Ingress
Write I/O response frame	Ingress	Egress	Ingress	Egress	Ingress	Egress

As is clear from Table 5-1 and Table 5-2, egress traffic on the host port, which is the same as the ingress traffic on the host-connected switchport, is due to:

- Read I/O command frames
- Write I/O command frames
- Write I/O data frames

Similarly, ingress traffic on the host port, which is the same as the egress traffic on the host-connected switchport, is due to:

- Read I/O data frames
- Read I/O response frames
- Write I/O transfer-ready frames
- Write I/O response frames

Typically, a network switch doesn't need to know the type of a frame (command, data, transfer-ready, or response frame) in order to send the frame toward its destination. However, without knowing the type of the frame, the real cause of throughput can't be explained. This is another reason for monitoring storage I/O performance by using SAN Analytics.

Network Traffic Throughput

The previous section explains the direction of traffic for read and write I/O operations. But not all the frames are of the same size. Read and write I/O data frames are large and usually occur in larger quantities. Hence, they are the major contributors to link utilization. Other frames, such as read and write I/O command frames, response frames, and write I/O transfer-ready frames, are small and relatively few. Hence, they cause much lower link utilization. Table 5-3 shows the typical sizes of different frame types for SCSI and NVMe I/O operations.

Table 5-3 *Typical Sizes of Frames for SCSI and NVMe I/O Operations*

FC Frame Type	FC Frame Size Using SCSI	FC Frame Size Using NVMe
Read command frame	68 bytes	68 bytes
Read data frame	I/O size of 2 KB or larger typically results in full-size FC frames (2148 bytes). Smaller I/O size operations result in smaller frame sizes.	I/O size of 2 KB or larger typically results in full-size FC frames (2148 bytes). Smaller I/O size operations result in smaller frame sizes.
Read response frame	60 bytes	60 bytes

FC Frame Type	FC Frame Size Using SCSI	FC Frame Size Using NVMe
Write command frame	68 bytes	132 bytes
Write transfer-ready frame	48 bytes	48 bytes
Write data frame	I/O size of 2 KB or larger typically results in full-size FC frames (2148 bytes). Smaller I/O size operations result in smaller frame sizes.	I/O size of 2 KB or larger typically results in full-size FC frames (2148 bytes). Smaller I/O size operations result in smaller frame sizes.
Write response frame	60 bytes	68 bytes

Correlating I/O Operations, Traffic Patterns, and Network Congestion

The directions and sizes of various frames in a storage network lead to the following conclusions:

- Read and write data frames are the major cause of link utilization. Other frames, such as command frames and response frames, are small, and their throughput is negligible compared to that of data frames.
- Read and write data frames flow only after (or as the result of) command frames.
- A command frame, based on the size of the requested data (called I/O size), can generate many data frames.
- Most data frames of an I/O operation are full sized, except the last frame in the sequence.
- Read data frames flow from storage (target) to hosts (initiators), whereas write data frames flow from hosts to storage.
- When a host-connected switchport is highly utilized in the egress direction, it's mostly due to read data frames. Likewise, when a storage-connected switchport is highly utilized in the egress direction, it's mostly due to write data frames.
- The key reason for congestion due to slow drain from hosts and due to overutilization of the host link is the multiple concurrent large-size read I/O command frames from the host. In other words, the host is asking for more data than it can process or than can be sent to it on its link.
- The key reason for congestion due to slow drain from a storage port or due to overutilization of the storage link is the total amount of data being requested by the storage array via multiple concurrent write I/O transfer-ready frames. In other words, the storage array is asking for more data than it can process or than can be sent to it on its link.

These conclusions are extremely useful in understanding the reason for congestion caused by a culprit device or the effect of congestion on the victim devices. These conclusions also explain that host port or switchport monitoring can detect congestion, whereas storage I/O performance monitoring can give insights into why the congestion exists.

For example, Figure 5-11 illustrates congestion due to overutilization of the host links because of large-size read I/O operations. The host connects at 32 GFC. It initiates 5000 read I/O operations per second (IOPS), each requesting to read 1 MB of data from various targets. To initiate these I/O operations, the host sends 5000 command frames per second, each 68 bytes, which leads to the host port's egress throughput of 2.8 Mbps (5000 × 68 B × 8 bits per byte), which is the same as the ingress throughput on the host-connected switchport. Because the maximum data rate of a 32 GFC port is 28.025 Gbps, these command frames result in 0.01% utilization, which is negligible.

The targets, after receiving these command frames, send the data for every I/O operation in approximately 512 full-size frames (2048 bytes per frame). For 5000 IOPS, the targets send 2,560,000 frames/second (5000 × 512), each 2148 bytes (including the header). These data frames lead to a throughput of 44 Gbps (2,560,000 × 2148 bytes × 8 bits per byte). But the host can receive only 28.025 Gbps on the 32 GFC link. This condition results in congestion due to overutilization of the host link. The key point to understand is that the ingress utilization of the host-connected switchport is negligible, yet this minimal throughput results in 100% egress utilization. From the perspective of the network, these are just the percentage utilizations of the links. Only after getting insight into the I/O operations can the real reason for the link utilization be explained.

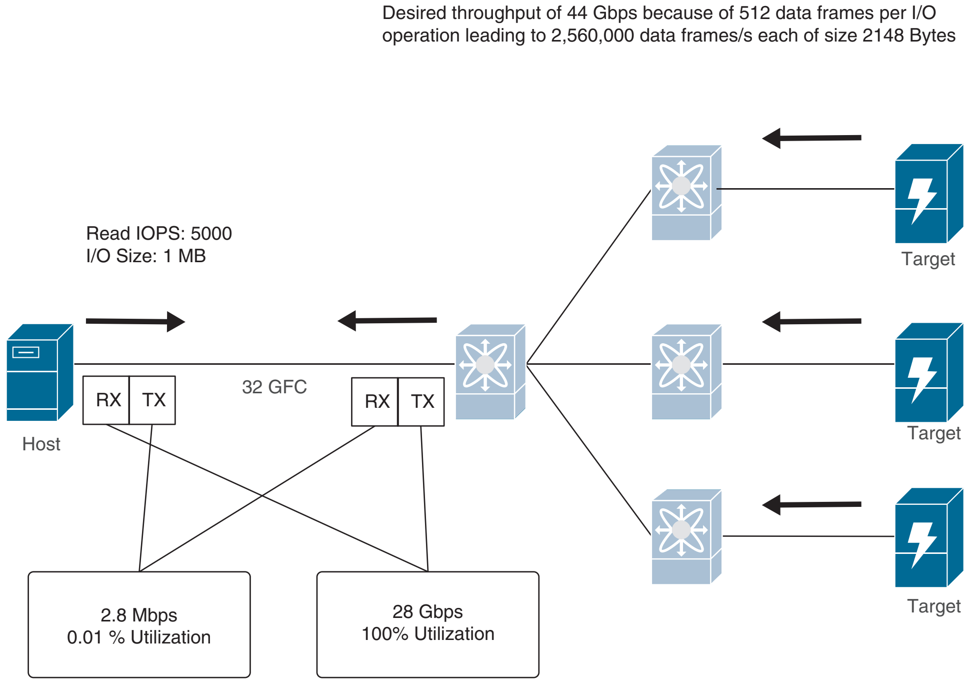

Figure 5-11 *Congestion Due to Overutilization Because of Large-Size Read I/O Operations*

Although the read I/O data frames make the most of the egress traffic on a host-connected switchport, these data frames are just a consequence of the read I/O command frames that were sent by the host port. Because limiting the rate of read I/O command frames can lower the rate of read I/O data frames, limiting the rate of ingress traffic on the host-connected switchport can lower the rate of egress traffic on this port. This logic forms the foundation of Dynamic Ingress Rate Limiting, which is a congestion prevention mechanism explained in Chapter 6, “Preventing Congestion in Fibre Channel Fabrics.”

Case Study 1: A Trading Company That Predicted Congestion Issues Using SAN Analytics

A trading company has thousands of devices connected to a Fibre Channel fabric, and it has multiple such fabrics. Because of the large scale, the company has always had minor congestion issues. However, the severity and number of such issues increased as the company deployed all-flash storage arrays. In an investigation, they found that the newer congestion issues were due to the overutilization of the host links. Most hosts were connected to the fabric at 8 GFC. The older storage arrays were connected at 16 GFC. But the newer all-flash arrays were connected at 32 GFC, which increased the speed mismatch between the hosts and the storage. As explained in Chapter 1, “Introduction to Congestion in Storage Networks,” this speed mismatch, combined with the high performance of all-flash arrays, was the root cause of the increased occurrences of congestion issues.

The trading company understood the problem and its root cause. It also understood that the real solution was to upgrade the hosts because doing so would eliminate the speed mismatch with the all-flash storage arrays, essentially removing one major cause of congestion due to overutilization of the host links. But, due to finite human resources, the company could only upgrade a few hundred hosts every month. At this pace, it would take many years to upgrade all the hosts, and the company would be subjected to congestion issues during this time. While the company could not speed up this change, it wanted to have a prioritized list of the hosts that were most likely to cause congestion. Instead of upgrading a host randomly or in an order that didn’t consider the likeliness of congestion, following this methodology would allow the company to minimize congestion issues.

Background

The trading company uses storage arrays from two major vendors. The hosts include almost all kinds of servers (such as blade and rack-mount servers) from all major vendors. The company uses all major operating systems for hosting hundreds of applications.

The trading company uses Cisco MDS switches (mostly modular directors) in its Fibre Channel fabrics. Most connections were capable of running at 16 GFC. However, while deploying all-flash arrays, they upgraded the storage connections to 32 GFC. For management and monitoring of the fabric, the company uses Cisco Data Center Network Manager (DCNM), which has since been rebranded as Nexus Dashboard Fabric Controller (NDFC).

Initial Investigation: Finding the Cause and Source of Congestion

The trading company used the following tools for detecting and investigating congestion issues:

- **Alerts from Cisco MDS switches:** The company had enabled alerts for Tx B2B credit unavailability by using the TxWait counter and alerts for high link utilization by using the Tx-datarate counter. As the company deployed all-flash arrays, the number of alerts generated due to TxWait didn't change, but the number of alerts due to Tx-datarate increased.
- **Traffic trends, seasonality, and peak utilization using DCNM:** After receiving the alerts from the MDS switches, the trading company used the historic traffic patterns in DCNM. The host ports that generated Tx-datarate alerts showed increased peak utilization. This increased utilization coincided with the time when the company deployed all-flash storage arrays.

These two mechanisms are explained in detail in Chapter 3, "Detecting Congestion in Fibre Channel Fabrics."

A Better Host Upgrade Plan

The trading company designed the host upgrade plan using two steps:

Step 1. Detect the hosts that were already causing congestion and upgrade them first.

Step 2. Predict what hosts were most likely to cause congestion and upgrade them next.

Step 1: Detect Congestion

The trading company detected the hosts that needed urgent attention, as explained earlier, in the section "Initial Investigation: Finding the Cause and Source of Congestion." These were the first ports to be upgraded, and the company prioritized upgrading the ports with slower speeds. But only a small percentage of the hosts made it to this list, and the company still wanted a prioritized list of the other hosts.

Step 2: Predict Congestion

The next step in designing a host upgrade plan (that is, a priority list of hosts) was finding the hosts that were most likely to cause congestion due to overutilization of their links.

In addition, the company wanted to find the hosts that were causing congestion but that could not be detected in Step 1. Any detection approach has a minimum time granularity. Events that are sustained for a shorter duration than the minimum time granularity often remain undetected. For example, even if congestion is detected at a granularity of 1 second, many congestion issues that are sustained for microseconds (sometimes called *microcongestion*) can't be detected. This is common with the all-flash storage arrays that have response times in microseconds. Because of this, the usual detection mechanisms used in Step 1 can't predict the likelihood of congestion.

This is where the insights obtained by using SAN Analytics help. The trading company enabled SAN Analytics on all its storage ports. Although only the storage ports inspected the traffic, the visibility from SAN Analytics was end-to-end at a granularity of every initiator, target, and logical unit (LUN) or ITL flow.

After collecting I/O flow metrics for a week, the company took the following steps (see Figure 5-12):

Step 1. The company extracted the read I/O size, write I/O size, read IOPS, and write IOPS for all the hosts.

Step 2. The company made sorted lists of the hosts according to read I/O size and read IOPS. In other words, the company found the hosts with the largest read I/O size and highest read IOPS. Write I/O size and write IOPS were not considered because, as mentioned in the section "Correlating I/O Operations, Traffic Patterns, and Network Congestion," most traffic due to write I/O operations flows from hosts to targets and does not lead to congestion due to overutilization of the host link.

Step 3. The company assumed that the hosts at the top of the list were more likely to cause congestion of their links and upgraded these hosts before upgrading the hosts with smaller read I/O sizes and lower IOPS.

Hosts	Peak I/O Size
Host-100	4 MB
Host-401	4 MB
Host-222	1 MB
Host-110	512 KB
Host-56	512 KB
…	…
…	…
Host-40	1 KB
Host-31	1 KB
Host-22	512 B
Host-5	512 B

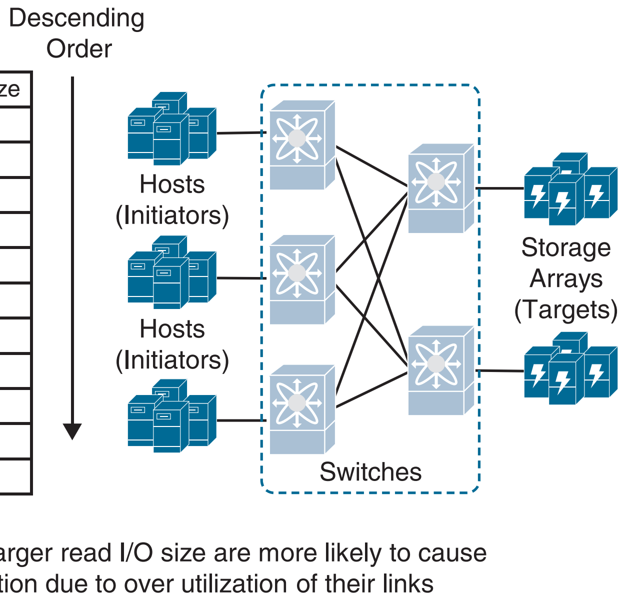

Figure 5-12 *Sorted List of Hosts Based on Peak Read I/O Size for Predicting Congestion Due to Overutilization*

A key consideration in predicting congestion is to focus on the peak values instead of the average values of the I/O flow metrics. This is because high average values indicate that the real-time values are sustained for a while. In this case, sustained traffic could have been detected by the Tx-datarate alert in Step 1, which has a granularity of 10 seconds.

But the Tx-datarate counter could miss occasional spikes in traffic that are sustained only for a few milliseconds or even seconds. Such conditions can be found or even predicted by focusing on the peak values of the I/O flow metrics.

Another consideration is to prioritize the I/O size metric over the IOPS metric—for two key reasons. First, as explained earlier in this chapter, in the section “I/O Size,” I/O size is determined by the application or the host, and it is not affected by network congestion. In contrast, IOPS is reduced during network congestion. The second reason is that I/O size is an absolute metric, which means it is directly collected from the frame headers. As a result, its peak value is not affected by averaging. In contrast, IOPS is a derived metric from the average number of I/O operations over a duration such as 30 seconds. Even the most granular value of IOPS must be calculated over a duration, which makes it an average value. This goes against the benefit of the peak values explained earlier.

For collecting data, the trading company used a custom-developed collector that polled the metrics for initiator flows every 30 seconds from the MDS switches and then used the peak values in 6-hour ranges. It was a custom development because this use case was very specific, and it was unavailable ready-made at that time on the MDS switches or SAN Insights. The raw metrics were available, but they were not available in an easy-to-interpret format. The custom development gave the company the easy-to-interpret format it wanted. This enhancement was later integrated with Cisco NX-OS running on MDS switches and it is available by default.

Example 5-2 shows the output of a similar custom development that is based on the **ShowAnalytics** command on MDS switches. It shows a sorted list of initiators according to their read I/O sizes. The **ShowAnalytics** command is a presentation layer for the raw flow metrics, and it is written in Python. Many use cases are available ready-made, and their functionality can be enhanced even further by users. More details are available at https://github.com/Cisco-SAN/ShowAnalytics-Examples/tree/master/004-advanced-top-iosize. Example 5-2 shows a modified version of the **ShowAnalytics** command.

Example 5-2 *Finding I/O Sizes of Hosts by Using SAN Analytics*

```
MDS# python bootflash:analytics-top-iosize.py --top --key RIOSIZE

+--------+----------------------------------------+------------------+
|  PORT  |        VSAN|Initiator|Target|LUN       |      IO SIZE     |
+--------+----------------------------------------+------------------+
|        |                                        |   Read  |  Write |
| fc1/35 | 20|0x320076|0x050101|002c-0000-0000-0000 |  1.2 MB |32.0 KB |
| fc1/34 | 20|0x320076|0x050041|000c-0000-0000-0000 |  1.1 MB |32.0 KB |
| fc1/33 | 20|0x320076|0x050021|002f-0000-0000-0000 |  1.0 MB |25.6 KB |
| fc1/35 | 20|0x320076|0x050101|001b-0000-0000-0000 |  1.0 MB |48.0 KB |
| fc1/33 | 20|0x320076|0x050021|0017-0000-0000-0000 | 992.0 KB|27.4 KB |
| fc1/33 | 20|0x320076|0x050021|0026-0000-0000-0000 | 992.0 KB|32.0 KB |
| fc1/33 | 20|0x320076|0x050021|0022-0000-0000-0000 | 960.0 KB|32.0 KB |
| fc1/34 | 20|0x320076|0x050041|0025-0000-0000-0000 | 960.0 KB|28.0 KB |
| fc1/35 | 20|0x320076|0x050101|001a-0000-0000-0000 | 960.0 KB|32.0 KB |
| fc1/34 | 20|0x320076|0x050041|0014-0000-0000-0000 | 928.0 KB|32.0 KB |
+--------+----------------------------------------+------------------+
```

Case Study 1 Summary

The trading company reduced its congestion issues by designing a two-step host upgrade plan. In Step 1, the company used the congestion detection capabilities of Cisco MDS switches and DCNM (NDFC). In Step 2, it used the predictive capabilities of SAN Analytics. Instead of upgrading the hosts randomly, the company prioritized upgrading the hosts that were more likely to cause congestion based on the peak read I/O size values. By following this plan, the company lowered the severity of congestion, and the number of such issues was only a fraction of what it had been at the beginning of the upgrade cycle, when the company started deploying all-flash arrays.

Case Study 2: A University That Avoided Congestion Issues by Correcting Multipathing Misconfiguration

A university observed congestion issues in its storage networks. After enabling alerting on the MDS switches, the university concluded that the congestion was due to the over-utilization of a few host links.

The university monitored the read and write I/O throughput on these hosts by using the host-centric approach described earlier in this chapter, in the section "Storage I/O Performance Monitoring in the Host." The throughput reported by the operating system (Linux) was much lower than the combined capacity of the host ports. This led the university to believe that ample network capacity was still available.

The university wanted to know why these hosts caused congestion due to overutilization even though the I/O throughput was less than the available capacity. Finding the reason for the congestion would pave the way to a solution.

Background

The university used the Port-Monitor feature to automatically detect congestion and generate alerts on Cisco MDS switches. It also enabled SAN Analytics and exported the metrics to DCNM/NDFC SAN Insights for long-term trending and end-to-end correlation of the I/O flow metrics.

Investigation

The university measured the host I/O throughput at the operating system, which was the combined throughput, but it had not measured the per-path I/O throughput. This was important because its hosts were connected to the storage arrays via two independent and redundant Fibre Channel fabrics (Fab-A and Fab-B). Most of its hosts have two HBAs, each with two ports (for a total of four ports). The first port on both HBAs connects to Fab-A, whereas the second port on both HBAs connects to Fab-B (see Figure 5-13).

The university used SAN Analytics to find the throughput per path, which is also available in DCNM SAN Insights. It found that although the combined throughput reported by SAN Insights was the same as the throughput measured at the operating system, the per-path throughput was not uniformly balanced. The ports connected to Fab-A were up to four times more utilized than the ports connected to Fab-B. When the host I/O throughput spiked, the increase seen on the ports connected to Fab-A was up to four times more than the increase seen on the ports connected to Fab-B. During this spike, the ports connected to Fab-A operated at full capacity, while the ports connected to Fab-B were underutilized. This was the reason for congestion due to the overutilization of host links in Fab-A.

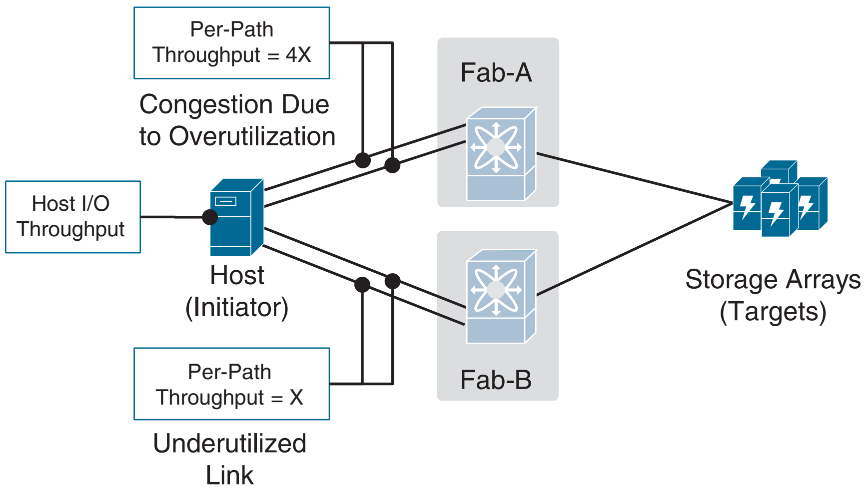

Figure 5-13 *Per-Path Throughput Monitoring Helps in Finding Multipathing Misconfiguration*

In Figure 5-13, traffic imbalance among the four host links can also be detected by measuring the utilization of host ports or their connected switchports. But if the hosts are within a blade server chassis, finding this traffic imbalance is not possible just by measuring port utilization. For example, in Cisco UCS architecture, the links that connect to the MDS switches can carry traffic for up to 160 servers, each with multiple initiators. Finding the throughput per initiator is possible only after getting flow-level visibility, as provided by SAN Analytics.

Figure 5-14 shows per-path throughput for the host and an end-to-end topology in DCNM/NDFC.

The root cause of this congestion was the misconfiguration of multipathing on these hosts. The university solved this congestion issue by correcting the multipathing misconfiguration on these hosts. SAN Analytics played a key role in finding the root cause because it was able to show a host's combined throughput as well as the per-path throughput.

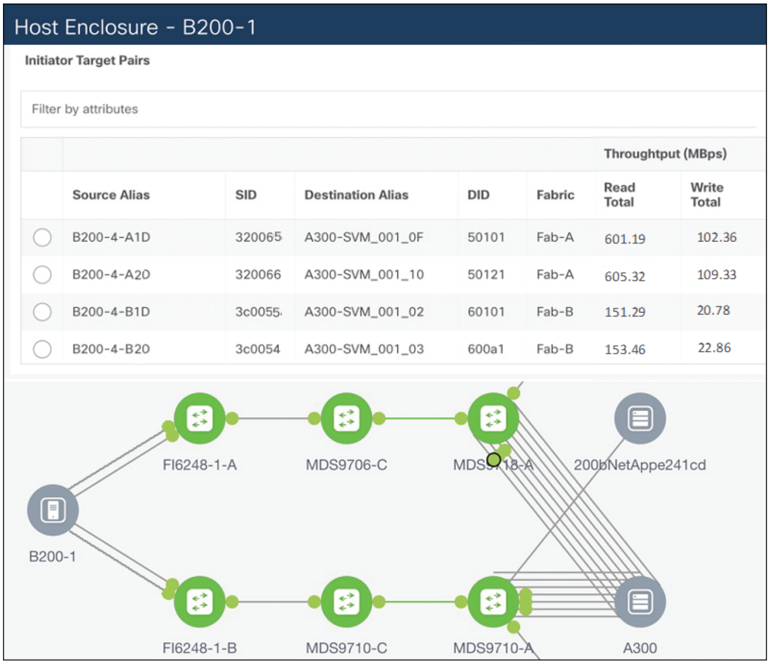

Figure 5-14 *Ready-Made View of the per-Path Throughput of Hosts in NDFC/DCNM SAN Insights*

Case Study 2 Summary

Using SAN Analytics, a university was able to find non-uniform traffic patterns that led to congestion due to overutilization of a few links while other links were underutilized. The insights provided by SAN Analytics pinpointed a problem at the host multipathing layer. The university solved the congestion issues by correcting the multipathing misconfiguration, which resulted in uniform utilization of the available paths.

Case Study 3: An Energy Company That Eliminated Congestion Issues

An energy company observed high TxWait values on its storage-connected switchports, which means the storage arrays had a slower processing rate than the traffic being delivered to them (that is, slow drain). Thus, the storage ports slowed down the sending of R_RDY primitives, leading to zero remaining-Tx-B2B-credits on the connected switchports, which led to high TxWait values.

The company observed the high TxWait values across all of its storage ports. No specific storage array stood out. Also, the TxWait spikes were observed throughout the peak business hours. The company couldn't pinpoint the high TxWait values to any specific hour.

The energy company wanted to know the reason for the high TxWait values on its storage-connected switchport. Knowing the root cause of this problem would allow them to find a solution before the issue became a business-impacting problem.

Background

The energy company uses storage arrays from a few major vendors. Its hosts include almost all kinds of servers (such as blade and rack-mount servers) from all major vendors. Most of its servers are virtualized using a leading hypervisor. The company uses Cisco MDS switches in its Fibre Channel fabrics. It used the Port-Monitor feature to automatically detect congestion and generate alerts for TxWait and other counters. However, not many alerts were generated because the TxWait values measured by the switchports were lower than the configured thresholds.

The energy company polls the TxWait value from all switchports every 30 seconds by using the MDS Traffic Monitoring (MTM) app (refer to Chapter 3). Cisco NDFC/DCNM Congestion Analysis also provides this information.

Investigation

The energy company needed more details to proceed with the investigation of high TxWait values on the storage-connected switchport because the existing data points were not conclusive. There were no specific time patterns or locations to pinpoint. TxWait values were observed throughout business hours randomly across all the storage-connected switchports. Also, some team members suspected issues within storage arrays. However, this possibility was ruled out because high TxWait values on the connected switchports were seen from all the storage arrays that had different vendors and different architectures.

The energy company took the following steps in investigating this issue:

Step 1. The company enabled SAN Analytics on the storage-connected switchports and allowed the I/O flow metrics to be collected for a week.

Step 2. Next, the company correlated TxWait values with ECT values on the storage ports. The ECT pattern matched with the TxWait pattern, which was expected because high TxWait values cause a delay in frame transmission, which in turn leads to longer exchange completion times.

Step 3. The company also tried matching the pattern of IOPS and throughput, but that didn't lead to any new revelations.

Step 4. The company correlated TxWait with I/O size. It didn't observe any matching patterns with read I/O size. However, it noticed that the time pattern of the spikes in write I/O size was an exact match with the time pattern of the spikes in TxWait.

Step 5. The company believed the spikes in write I/O size could explain the spikes in TxWait on the storage ports. It used this reasoning:

- Typically, the write I/O size was in the range 512 bytes to 64 KB. During the spikes, the write I/O size increased to 1 MB. A 64 KB write I/O operation results in 32 full-size Fibre Channel frames, and a 1 MB write I/O operation results in 512 full-size Fibre Channel frames.
- Most traffic due to a write I/O operation flows from hosts to storage ports.
- The spike in write I/O size caused a burst of frames toward the storage arrays.
- It was possible that the storage arrays could not process the burst of the frames in a timely manner and used the B2B flow control mechanism to slow down the ingress frame rate. The storage arrays reduced the rate of sending R_RDY primitives, leading to zero remaining-Tx-B2B-credits on the connected switchport, which led to high TxWait values.

Step 6. After determining that the large write I/O operations were the reason for the TxWait values on storage-connected switchports, the company wanted to resolve this issue. It had to find which hosts (initiators) and possibly which applications used the large-size write I/O operations.

Step 7. The company used SAN Analytics to find the write I/O size for every initiator-target-LUN (ITL) flow on the storage-connected switchports. This detailed information was enough to find the hosts (initiators) that initiated the large-size write I/O operations.

Step 8. Using SAN Analytics, the company found that these ITL flows had been active, and they had been doing write I/O operations with typical I/O sizes in the range 512 bytes to 64 KB. The write I/O size spiked to 1 MB just before these ITL flows stopped showing any I/O activity. In other words, the IOPS and throughput of these ITL flows dropped to zero right after the spike in write I/O size to 1 MB. It was an interesting pattern that was commonly seen on all the ITL flows that showed spikes in write I/O size to 1 MB.

Step 9. The company located the servers by using the initiator value from the ITL flows. Because these servers were virtualized, the company used the LUN value from the ITL flow to locate the datastore and a virtual disk on the hypervisor. However, it couldn't find any data store or a virtual disk that was associated with the LUN value.

Step 10. Because the data from SAN Analytics showed nonzero IOPS for the ITL flows, the company was confident that these hosts used the storage volume associated with the LUN. Initially, it thought that it was not seeing all the information from the hosts. But later it was suspected that probably all these hosts stopped using the LUN. Not using the LUN coincided with the traffic pattern where the ITL flows showed no I/O activity right after a spike in the write I/O size.

Step 11. The company suspected some cleanup mechanism before freeing up the disks. The application and virtualization teams found that, as per the company's compliance guidelines, explicit (eager) zeros are written before the volumes are freed up.

Step 12. The company found that many applications were short-lived. When such applications are provisioned, the company creates virtual machines and allocates storage. As soon as an application is shut down, the virtual machine resources are freed. During this process, the company wipes all the data and then writes (eager) zeros on the volumes.

Step 13. Next, the company found the disk cleanup process. The hypervisor documentation made it clear that this cleanup process of writing zeros used an I/O size of 1 MB. This value matched with the write I/O size value shown by SAN Analytics on the storage-connected switchport that reported spikes in TxWait values. This also explained why no I/O activity was seen right after the write I/O size spiked.

Step 14. The company concluded that the disk cleanup process was the root cause of the spikes in write I/O size, which in turn caused the spikes in TxWait values on the storage-connected switchports. To test this idea, the company followed the same sequence of deploying an application followed by shutting it down. When the virtual machine was freed, the company could match the timestamps on the hypervisor with the spike in write I/O size for the corresponding ITL flow on the storage port, as reported by SAN Analytics. Connecting these end-to-end dots between the storage network and the application gave the company a clear understanding of the root cause of the problem. However, the problem was not yet solved. Because of the compliance guidelines, the company couldn't stop the disk cleanup process. Also, changing the default write I/O size of the disk cleanup process was perceived to be risky.

Step 15. The company's final approach, which aligned with its compliance guidelines and was agreed upon by all the teams, was to avoid cleaning up the virtual machines during peak business hours. The company changed the workflow to not free up the virtual machine immediately after the application was shut down. Rather, it delayed the cleanup process until off-peak (late-night) hours.

Step 16. The company verified this change by using the TxWait values on switchports and write I/O size, as reported by SAN Analytics. It didn't see spikes in TxWait values anymore. It saw spikes in write I/O size, but now TxWait values didn't increase, probably because the overall load on the storage arrays was low during the off-peak hours, and thus, the spike of the write I/O size for some flows didn't cause processing delays with the storage arrays.

Figure 5-15 shows a TxWait graph in NDFC/DCNM Congestion Analysis. This graph has a granularity of 60 seconds. TxWait of 30 seconds in this graph translates to 50% TxWait.

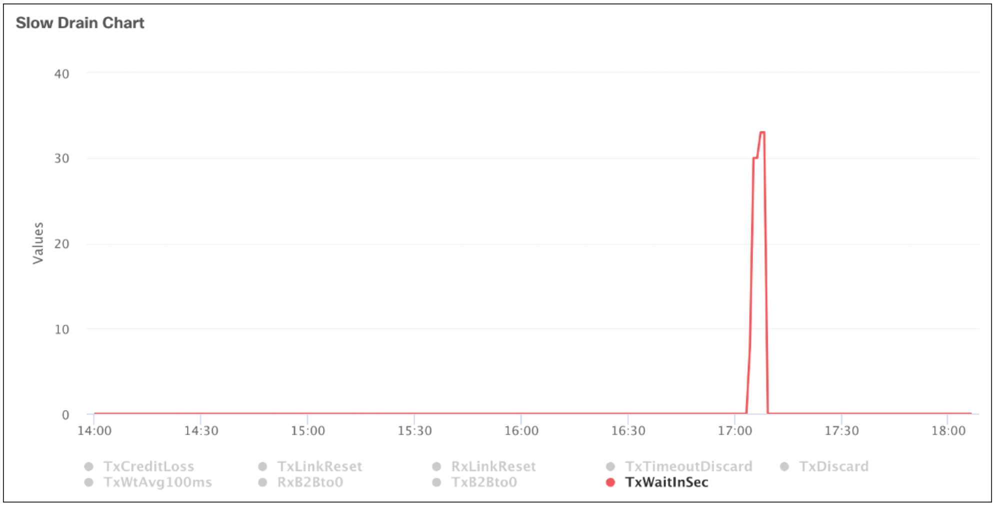

Figure 5-15 *TxWait in NDFC/DCNM Congestion Analysis*

Figure 5-16 shows a write I/O size time-series graph in NDFC/DCNM SAN Insights. Notice the sudden spike and timestamp.

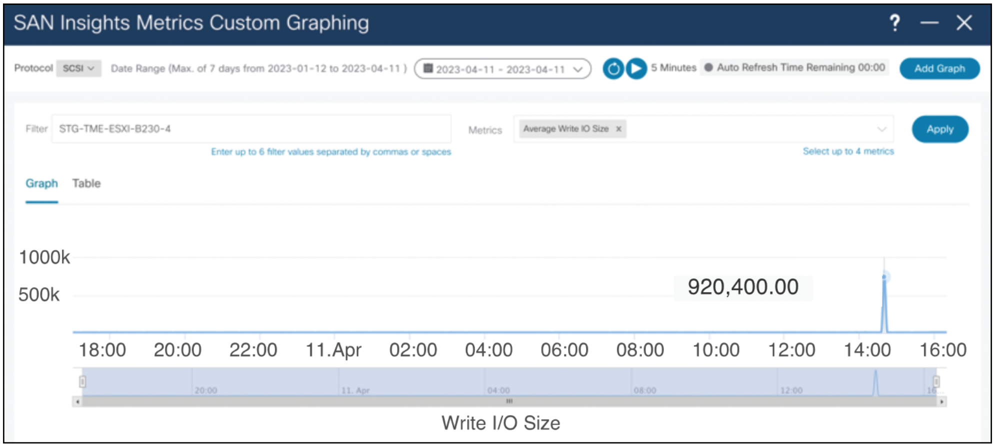

Figure 5-16 *Write I/O Size Spike in NDFC/DCNM SAN Insights*

Figures 5-15 and 5-16 are close representations, but they are not sourced from the environment of the energy company. They are shown here to illustrate how the spikes in TxWait values and I/O size can be found and used.

Case Study 3 Summary

Using SAN Analytics, the energy company was able to find the root cause of high TxWait values on the storage-connected switchports and eliminate this congestion issue. First, it found that the spike in TxWait values was caused by the spike in write I/O size. Then it found the culprit ITL flows and used the initiator and LUN values to locate the hosts and the virtual machine. Finally, it used the traffic pattern—zero I/O activity just after a spike in write I/O size—to conclude that the disk cleanup process was the root cause of the spike in write I/O size. Based on this conclusion, the company solved the problem by delaying the disk cleanup until off-peak hours. This simple step eliminated congestion (TxWait spikes) from the company's storage-connected switchports, which essentially led to better overall storage performance. This performance optimization wouldn't have been possible without the insights provided by SAN Analytics.

Case Study 4: A Bank That Eliminated Congestion Through Infrastructure Optimization

A bank had an edge–core design in a storage network that connects thousands of devices. It often received a high egress utilization alert from a switchport connected to Host-1. The high-utilization condition persisted for a few minutes, and it happened a few times every day. While this switchport reported high egress utilization, congestion was seen on the ISL ports, as confirmed using TxWait values on the ISL ports of the upstream switch.

The bank had a large server farm, and many servers were underutilized. It was believed that high egress utilization on the switchport connected to Host-1 could be eliminated by moving some of the workloads to another server. However, instead of randomly moving a workload to another server (which would be a hit-or-miss approach), the bank wanted to make a data-driven decision to make the right change in one attempt. Every change is expensive, and the cost multiplies quickly in large environments.

Background

The bank used storage arrays from a few major vendors. Its hosts deployment included almost all kinds of servers (such as blade and rack-mount servers). Most of its servers were virtualized using a leading hypervisor. The bank used Cisco MDS switches in its Fibre Channel fabrics. It had enabled automatic monitoring and alerting using the Port-Monitor feature on MDS switches.

Using the high egress utilization (Tx-datarate) alerts, the bank was able to find the following information:

- **When the congestion started:** This was based on the timestamp of the Port-Monitor alerts.
- **How long the congestion lasted:** This was determined by finding the difference in timestamps between the rising and falling threshold events.

- **Where the source of congestion was located:** Port-Monitor alerts reported which switch and switchport were highly utilized. The FLOGI database (via the NX-OS command **show flogi database**) showed that the affected switchport was connected to Host-1.
- **The congestion severity:** This was reported by the Tx-datarate counter on the switchport that connected Host-1 and TxWait on the ISL ports of the upstream switch (refer to Chapter 4, "Troubleshooting Congestion in Fibre Channel Fabrics").

Investigation

The bank needed more details to make a data-driven change to reduce the high ingress utilization of the Host-1 port, which is the same as the egress utilization of the connected switchport. Although the metrics from the switchport and the alerts from the Port-Monitor showed high utilization, granular flow level details were not available.

The bank wanted to move some workload from Host-1 to the other underutilized servers. But it didn't know which workload to move and to which server.

The bank went through the following steps in investigating this issue:

> **Note** For the sake of simplicity, this explanation limits the scope to only four servers (Host-1 through Host-4).

Step 1. The bank enabled SAN Analytics on the host-connected switchports and ran it for a week while the same pattern of overutilization and congestion repeated. This helped in collecting end-to-end I/O flow metrics.

Step 2. Using SAN Analytics, the bank found the number of targets (using IT flows) and the number of logical units (storage volumes, or LUNs) (using ITL flows) that each server was doing I/O operations with. Table 5-4 shows the findings.

Table 5-4 *Distribution of IT and ITL Flows of the Servers*

Server Name	Number of IT Flows	Number of ITL Flows	Number of LUNs (ITL flows / IT flows)
Host-1	4	40	10
Host-2	4	20	5
Host-3	4	12	3
Host-4	4	80	20

Dividing the number of ITL flows by the number of IT flows gave the bank the number of LUNs that each server was doing I/O operations with. The results indicated that Host-1 was accessing a higher number of LUNs than were Host-2 and Host-3. Host-4's LUN number was double that of Host-1, yet it didn't cause utilization as high as for Host-1.

Step 3. The bank found the throughput for every ITL flow. It focused on read I/O throughput because most egress traffic on host-connected switchports results from read I/O operations. After sorting the ITL flows on the Host-1 connected switchport as per the read I/O throughput, the bank found an ITL flow that had a throughput much higher than the other ITL flows. Also, the pattern of spikes and dips of the read I/O throughput of this ITL flow matched the egress utilization on the Host-1 connected switchport. Clearly, this ITL flow was the major cause of the high utilization of the switchport and, consequently, the reason for congestion on the ISL.

Step 4. The bank wanted to find the workload that was using this ITL flow. Host-1 was virtualized, with many virtual machines. The bank used the LUN value of the ITL flow to find the datastore. It found the virtual disk that was created using this datastore and found the virtual machines that were using that virtual disk. To verify that it had located the correct virtual machine, the bank used the I/O throughput as reported by the operating system of the VM and matched it with the throughput reported by SAN Analytics for the detected ITL flow.

Step 5. After locating the high-throughput virtual machine on Host-1, the bank wanted to find the best server to which this virtual machine could be moved. Was it Host-2, Host-3, or Host-4?

Step 6. The bank ruled out Host-4 because it already had a greater number of ITL flows. The remaining possible options were Host-2 with 20 ITL flows, and Host-3 with 12 ITL flows.

Step 7. The bank found more metrics reported by SAN Analytics. Table 5-5 shows these findings.

Table 5-5 *I/O Flow Metrics from SAN Analytics for Host-2 and Host-3*

Server Name	Peak Egress Utilization of the Connected Switchport	Peak IOPS	Peak Read I/O Size
Host-2	30%	10,000	16 KB
Host-3	40%	2000	64 KB

It was important to use the peak values in order to make the right decisions because congestion issues are more severe under peak load. Based on this data, the bank decided to move the high-throughput virtual machine from Host-1 to Host-2 because of its lower utilization and lower read I/O size. Had it made the decision based on the number of ITL counts alone, the bank would have chosen Host-3, which was not the best choice. By using the insights provided by SAN Analytics, the bank was able to make a data-driven decision.

The bank continued to monitor the servers and repeated these steps for further optimization.

Case Study 4 Summary

The bank received high egress utilization alerts from one of the host-connected switch-ports, which led to congestion on the ISL. It resolved this issue by moving a high-throughput workload/VM from this host to other underutilized hosts. To make this change, the bank used SAN Analytics to find the number of IT flows and ITL flows. It then found the throughput per flow and sorted the flows according to throughput to find the culprit flow. Next, the bank located the virtual machine by using the LUN value from the ITL flow and correlated it with the datastore and virtual disk on the hypervisor. Finally, it analyzed the peak throughput, IOPS, and I/O sizes of the other servers to find the best host for the high-throughput workload.

The insights provided by SAN Analytics helped the bank resolve this issue with only one change.

Summary

Storage I/O performance monitoring provides advanced insights into network traffic, and these insights can be used to accurately solve network congestion. Cisco SAN Analytics, which takes a network-centric approach to storage I/O performance monitoring, provides end-to-end visibility into I/O operations between virtual machines, initiators, targets, and LUNs/namespaces. The per-flow performance metrics from SAN Analytics help in determining network traffic patterns. For example, the throughput on a port can be predicted by using the I/O size of the read and write operations. Also, most throughput due to read I/O operations is in the direction from storage (target) to hosts (initiators), whereas most throughput due to write I/O operations is in the direction from hosts to storage. Although the read and write I/O data frames make the most of the traffic, these data frames are just a consequence of the read and write I/O command frames that are sent from the hosts to the target. These details help in detecting and predicting congestion issues, and they also help in preventing them by using mechanisms like Dynamic Ingress Rate Limiting, as explained in Chapter 6.

This chapter explains the practical usage of SAN Analytics via four case studies. The steps explained in these case studies can be reused in other environments for detecting and predicting congestion issues.

Finally, storage I/O performance monitoring and SAN Analytics are detailed subjects, and these tools can achieve a lot more than detecting and predicting congestion in storage networks. We recommend continuing your education on this topic outside this book.

References

Cisco SAN Analytics and SAN Telemetry Streaming Solution Overview, https://www.cisco.com/c/en/us/products/collateral/storage-networking/mds-9700-series-multilayer-directors/solution-overview-c22-740197.html

Cisco MDS 9000 Series SAN Analytics and SAN Telemetry Streaming Configuration Guide, https://www.cisco.com/c/en/us/td/docs/dcn/mds9000/sw/9x/configuration/san-analytics/cisco-mds-9000-san-analytics-telemetry-streaming-configuration-guide-9x.html

DCNM SAN Insights, "Next Generation Network Visibility," BRKDCN-2271, Cisco Live 2019, San Diego.

DCNM SAN Insights, "Next Generation Network Visibility," BRKDCN-3645, Cisco Live 2022, Las Vegas.

"Detecting, Alerting, Identifying, and Preventing SAN Congestion," BRKDCN-3241, Cisco Live 2022, Las Vegas.

"SAN Congestion: Understanding, Troubleshooting, Mitigating in a Cisco Fabric," BRKSAN-3446, Cisco Live 2017, Las Vegas.

ISO/IEC 14165-226:2020, Fibre Channel Single-Byte Command Code Sets Mapping Protocol–6 (FC-SB-6)

IANA, Service Name and Transport Protocol Port Number Registry, https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

NVMe over Fabrics Specification, http://www.nvmexpress.org

NVM Express Base Specification, Revision 2.0, http://www.nvmexpress.org

INCITS 514-2014, Information Technology: SCSI Block Commands–3 (SBC-3), http://webstore.ansi.org

NVM Express RDMA Transport Specification, Revision 1.0, https://www.nvmexpress.org

NVM Express TCP Transport Specification, Revision 1.0, https://www.nvmexpress.org

Index

A

B

C

D

E

F

G

H

I

J - K - L

M

N

O

P

Q

R

S

T

U

V

W

X - Y - Z