Detecting, Troubleshooting, and Preventing Congestion in Storage Networks

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

ciscopress.com

Contents

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.

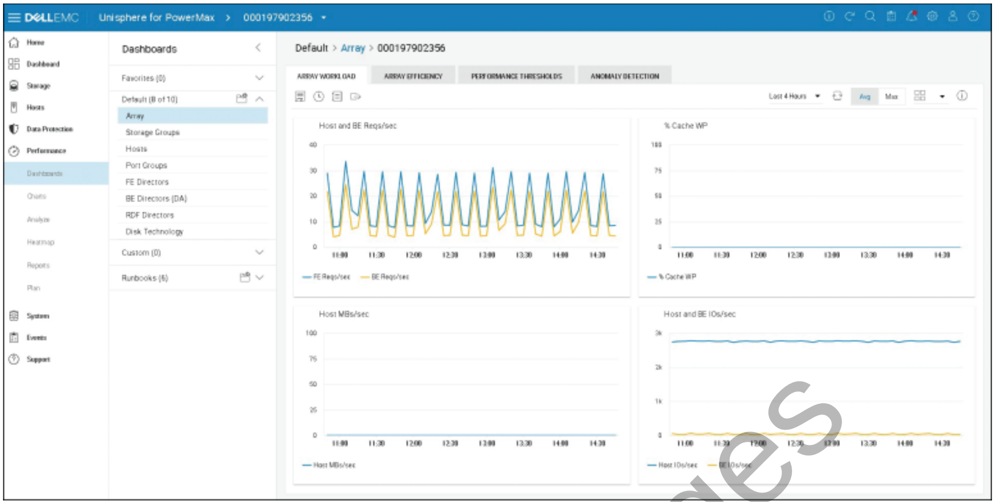

Figure 5-1 *Storage I/O Performance Monitoring on a Dell EMC PowerMax Storage Array*

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

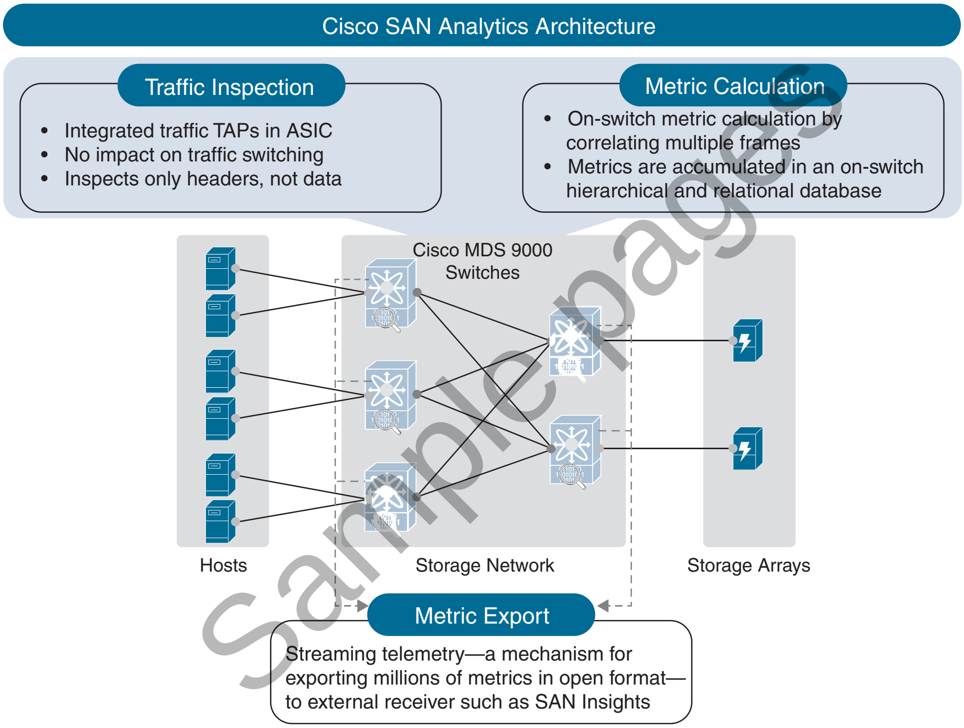

Figure 5-2 *Cisco SAN Analytics Architecture*

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.

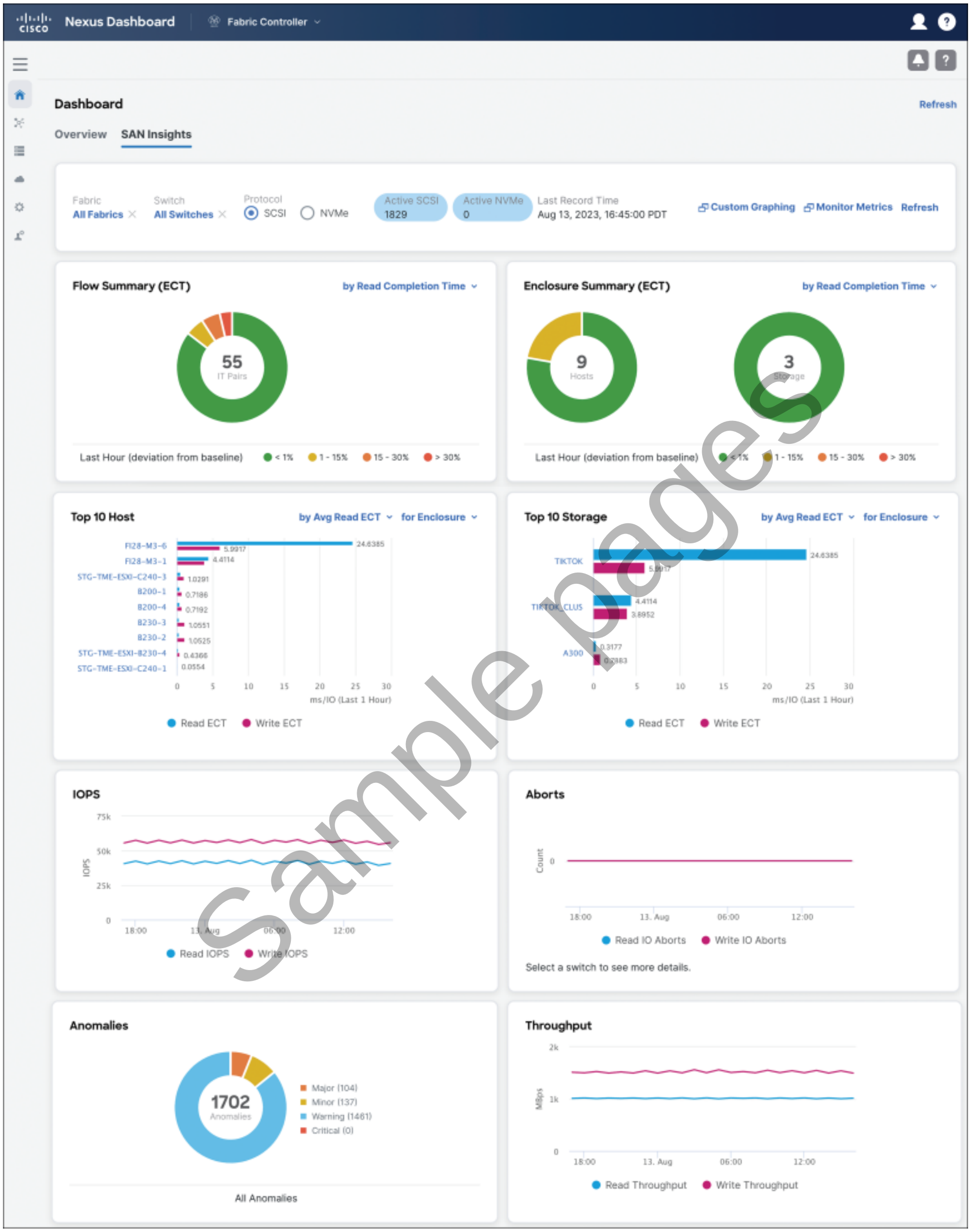

Figure 5-3 *SAN Insights Dashboard in Cisco NDFC*

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.

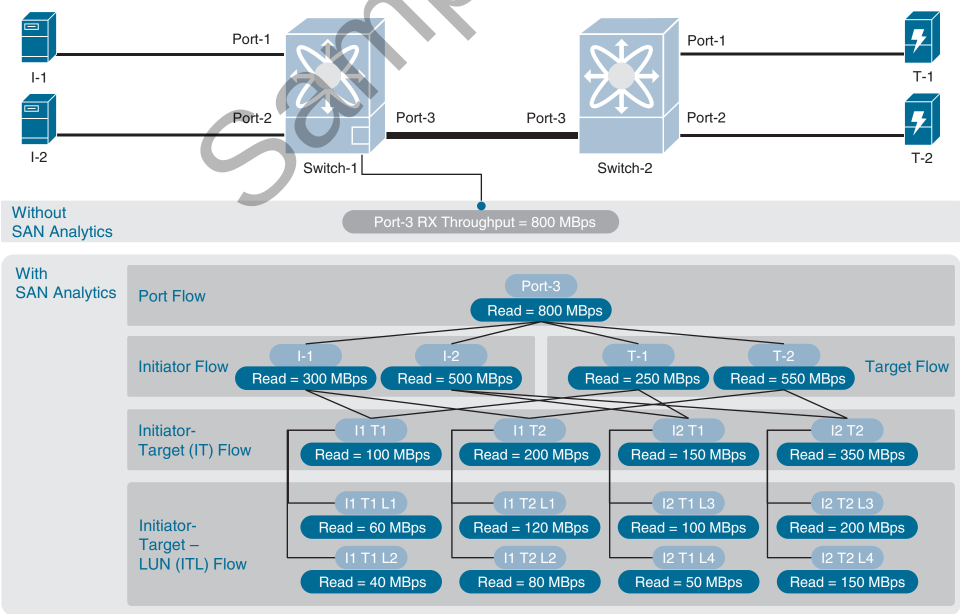

Figure 5-4 *I/O Flows and Flow-Level Metrics Using Cisco SAN Analytics*

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).

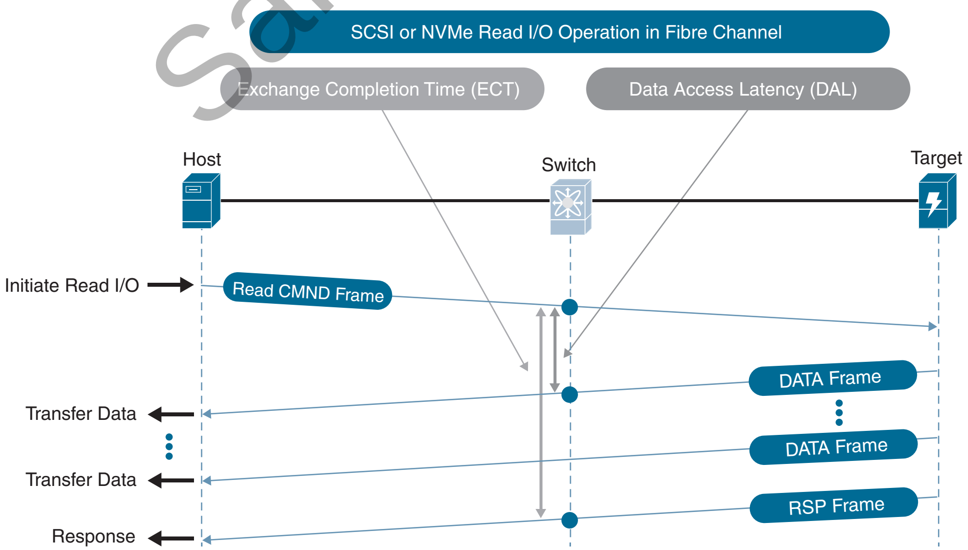

Figure 5-5 *Latency Metrics for a Read I/O Operation*

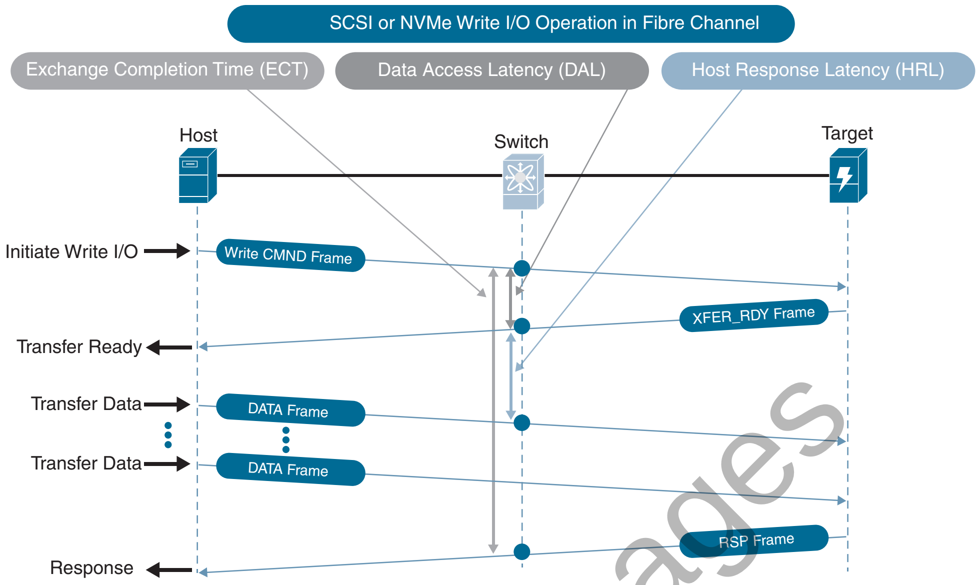

Figure 5-6 *Latency Metrics for a Write I/O Operation*

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.

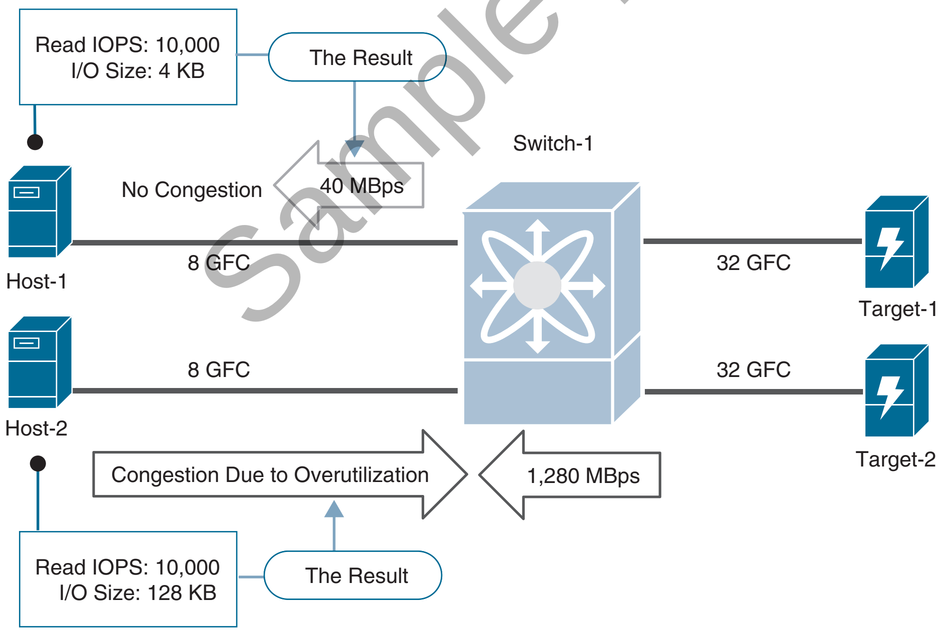

Figure 5-7 *Detecting and Predicting the Cause of Congestion Using I/O Size*