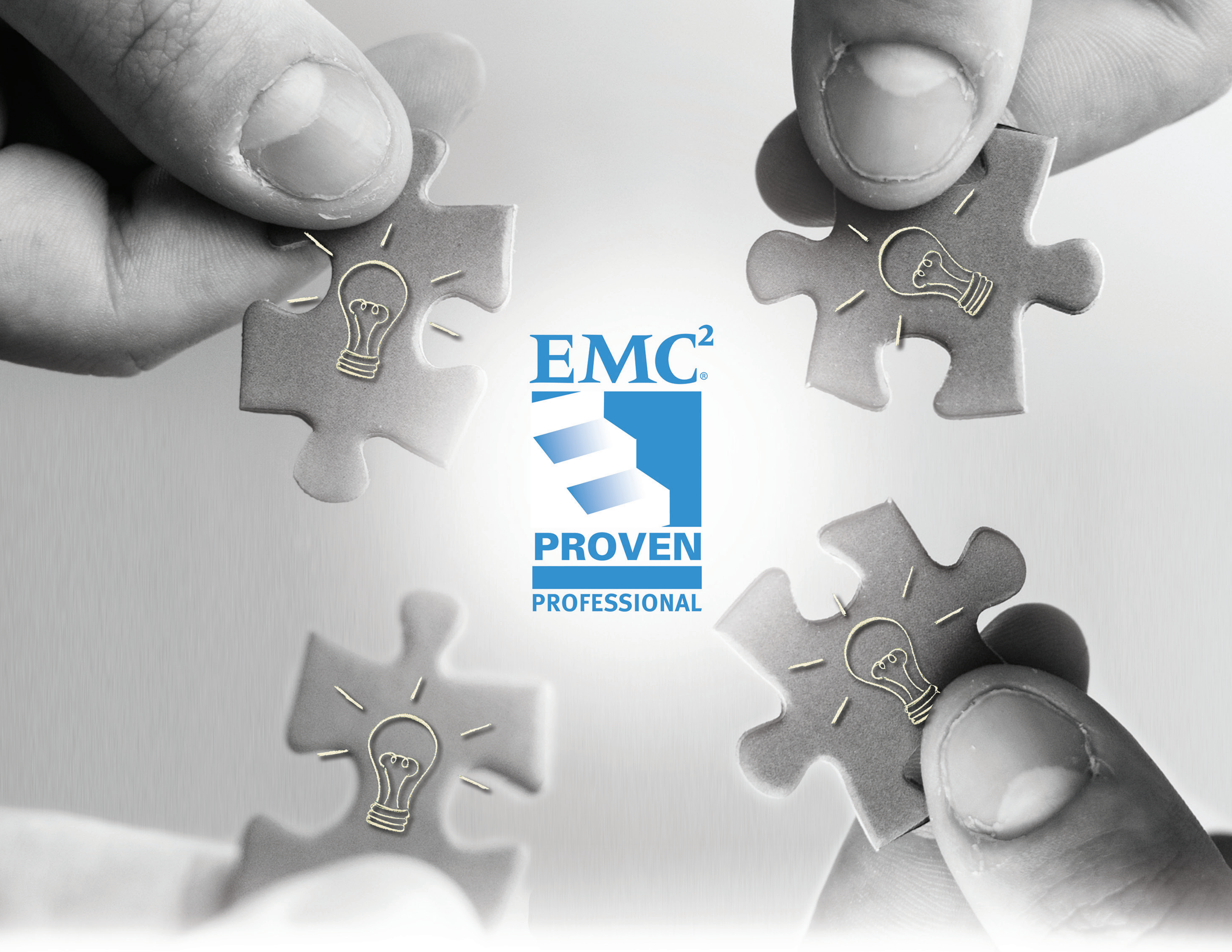


UNDERSTANDING HOW I/O WORKLOAD PROFILES RELATE TO PERFORMANCE

Stan Lanning
Principal Systems Engineer
Enterprise Storage Division Specialist
EMC Corporation
stan.lanning@emc.com

EMC²

Table of Contents

Introduction

When implementing a new application there is usually a requirements section or sizing guide that indicates minimum number of server processors or cores, the amount of memory, and storage capacity required. In more complex application systems such as Microsoft Exchange, SAP, or Oracle E-Business Suite there may be sizing calculators to help determine the number of servers, network bandwidth, database requirements, and so on. However, there are relatively few tools for storage performance sizing, and they tend to be specific to individual products or vendors. Plus, the inputs to these tools are often not well understood or simply not known.

As a storage solutions specialist involved in sizing IT infrastructure, one of the first questions I ask customers and sales teams is, “what are the IO workload profiles that the infrastructure needs to support?” Often, I receive a response of “100 Terabytes”, “2700 IOPS”, “350 Megabytes per second”, or simply “we don’t know”. Then I ask a series of questions trying to understand what applications the customer is running or planning to run, how data flows in and out of the system, and hopefully gather enough detail to construct the “IO workload profiles”. This information is necessary to design a storage solution that will meet performance and cost requirements.

What are “IO workload profiles”?
Why are they important?
How do we identify them?
How do they relate to performance sizing?

This article will explain the fundamentals of the characteristics of IO workload profiles – such as IOPS, MB/s, read/write ratios, response times, and queue lengths. It will also explore how different profiles can impact performance sizing, and some common tools that can be used to identify workload profile characteristics.

Why is an understanding of workload profile important?

If we do not understand the workload profile there is significant risk that the storage solution will not meet desired performance levels which can slow or even halt a business function. Or the solution may be “over sized” leading to higher than necessary acquisition, implementation, and operating costs.

Key Storage Workload Characteristics

There are many characteristics of an IO workload that affect performance – in fact, too many to cover in this article. We will focus on several key characteristics that describe and affect all storage devices, including:

- IO Size
- IOPS
- Throughput
- Response Time
- Read/Write Ratio
- Random vs. Sequential

A future article will go into more depth on additional aspects of workload profiles including “locality of reference”, “cache friendliness”, “burstiness”, and so forth

IO Size

In computing systems, the size of the Input or Output request (IO size) is typically expressed in Bytes (B), Kilobytes (KB), or sometimes Megabytes (MB).

The smaller an IO, the less time it takes to transmit. Similarly, the larger an IO, the longer it takes to transmit. However, there are advantages to using small and large IO sizes depending on the type of work being performed. Using too small or too large a size can lead to additional complications.

Here is an example:

> Imagine your parents are moving to a new house and you need to move your collection of vintage comic books out of their basement before they are thrown away. Each comic book is similar to a file containing text and media content.
>
> If you take one comic book at a time, there is little effort required to move each one and you can quickly move it from basement to the trunk or boot of your waiting car. However, you will have to make many trips to move the entire collection. This is similar to using a small IO size – very quick to move, but lots of round trips to complete the work.
>
> If you pack 80 comic books into a large box and then move the box, it takes longer to pack, movement is slower due to the heavy box, and navigating the narrow basement

staircase may be more difficult. But the overall move may go faster – this is similar to large IO sizes – because lots of comic books are moved in each trip.

In most situations, the most efficient method involves packing the comic books into several medium size boxes because there will be fewer round trips than taking one comic book at a time, but you also avoid the long packing time and difficult maneuvering with a very large box. This is the compromise that most applications need to trade-off – moving data efficiently in the right size containers with as few round trips as possible.

Consider another example:

A roadway provides passage for cars and trucks of different sizes. In most countries the roadway or each lane has a standard width. Vehicles must comply with width and height limits in order to avoid obstacles such as surrounding traffic, overhead wires, bridges, and so on. Length is the primary variable in determining how much stuff – people and/or cargo – a vehicle can transport.

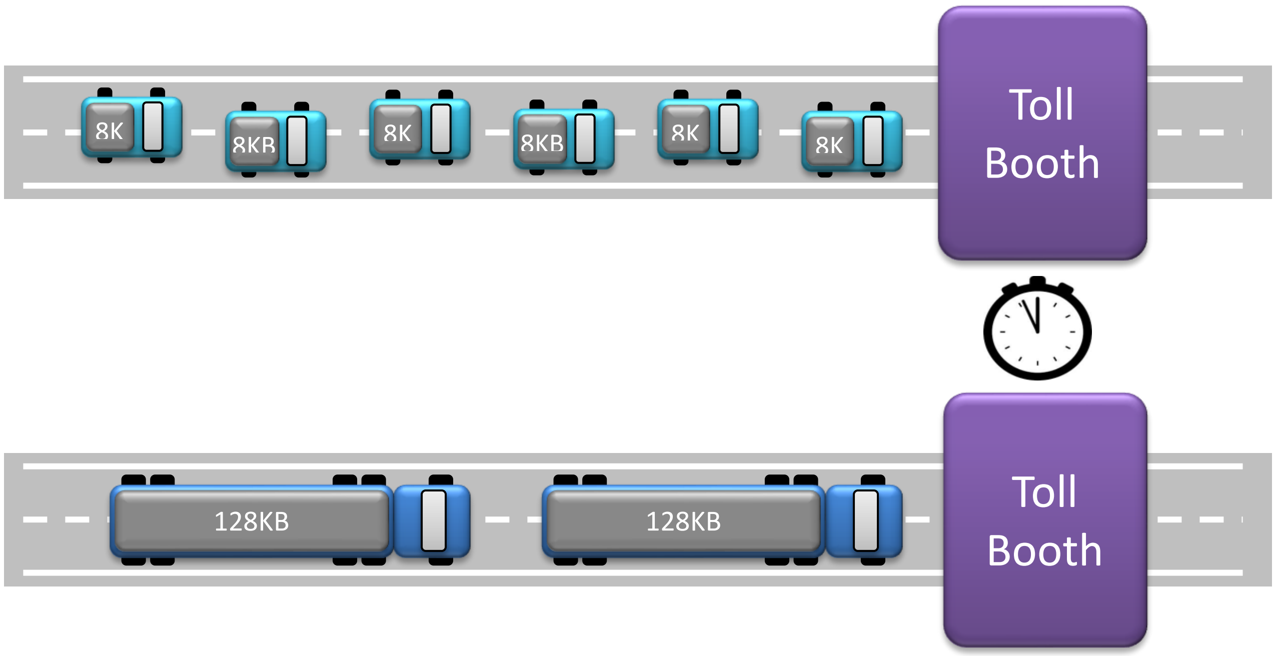
A car is generally short in length, width, and height – easier to maneuver and economical to operate, but does not hold very much cargo. A large truck is wider, taller, and much longer than most cars, and it can carry many times the amount of cargo.

The size of the vehicle does not necessarily determine how fast it can go. Most vehicles today can travel faster than the posted speed limit. Also keep in mind the size of the vehicle often affects how much work (fuel) is required to move the vehicle, but for now the analogy will avoid the complexities of acceleration, friction, and so on.

IOPS

IOPS measures the number of **I**nput / **O**utput operations (or requests) **P**er **S**econd. This is sometimes referred to as requests per second.

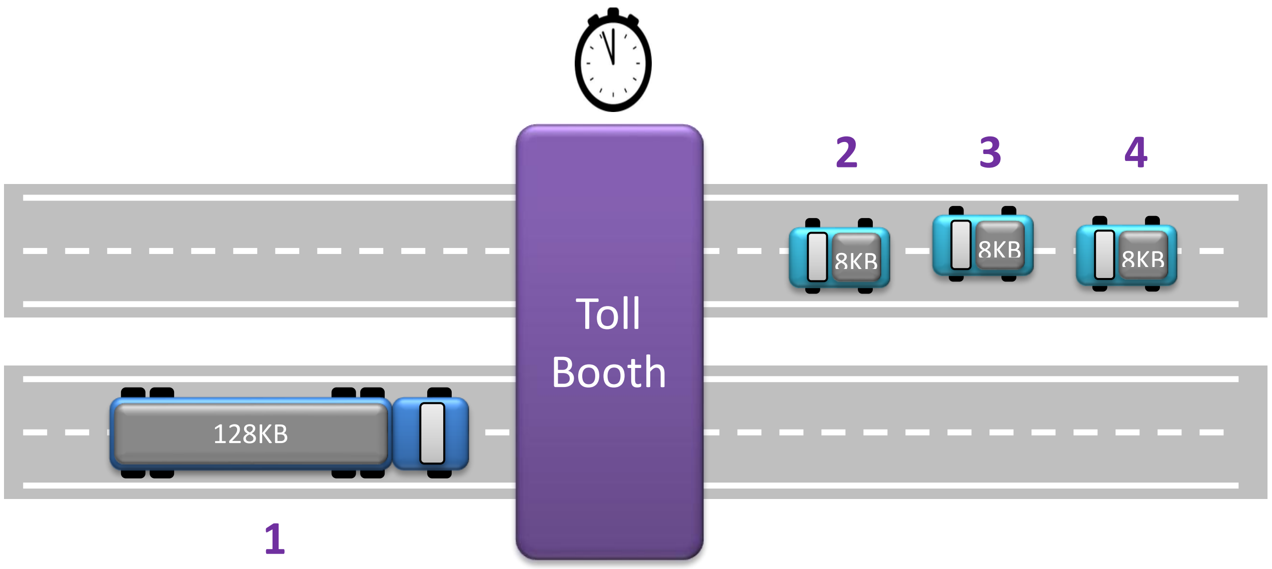
Consider a road with cars and trucks going through toll booths. Assuming all vehicles are traveling at the same speed, the number of small cars that pass through the toll plaza in a given period of time will be much higher than the number of large trucks that pass through the toll plaza – mostly due to the length of the vehicle. Plus, the amount of cargo that passes through the toll booth will vary even more due to the carrying capacity of cars vs. trucks.



The number of vehicles that pass through the toll booth in one hour is analogous to the number of bytes that pass through an interface each second – IOPS.

Most roads have vehicles traveling in two directions, and the number of vehicles going in either direction is similar to the total IOPS of a storage or network interface. However, we may want to know how many vehicles travel in one particular direction in a given time period.

Four vehicles passing through within one second equals 4 IOPS – regardless of which direction they are traveling.



In network and storage environments, we often need to know the number of read and write *requests (inputs and outputs) per second as separate values.*

IOPS may be expressed together – total IOPS, or separately as read or write IOPS:

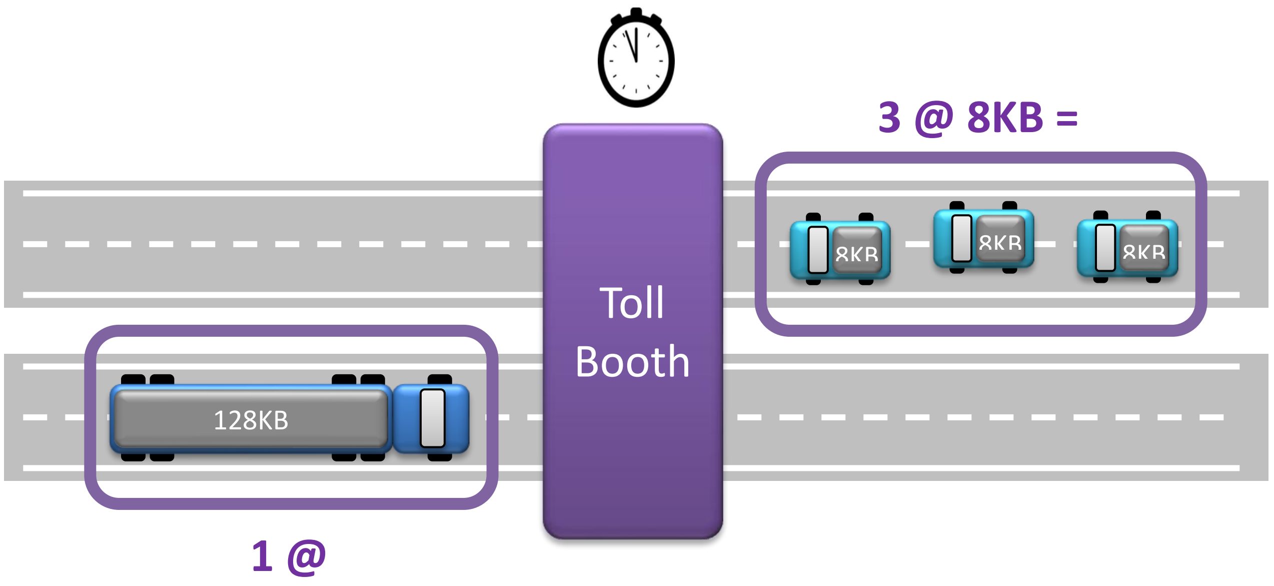
- IOPS = combined reads + writes per second
- rIOPS (or r/s) = reads per second
- wIOPS (or w/s) = writes per second

Throughput

Throughput measures the amount of work performed in a given time period.

Returning to the toll road example – while IOPS measures the number of vehicles that pass through the toll booths in an interval, throughput measures the amount of cargo that passes through the toll booths in a given period of time.

Assume the large truck and three small cars pass through the toll booth within one second – the total cargo is 128KB + 8KB + 8KB + 8KB within one second, or a throughput of 152KB/s.



In computer network and storage environments, throughput –sometimes called bandwidth - refers to the amount of data processed in one second, minute, or hour.

Here we introduce the critical three-way relationship - ***IOPS * IO Size = Throughput***. For example:

- An application processing 100 requests per second where each request is 4KB in size has a total throughput of 400KB/s
  - 100 IOPS * 4KB = 400KB/s
- A server with four HBAs – where each HBA processes 1,200 reads and 700 writes per second with an average IO size of 32KB – has an average throughput of 243,200KB/s
  - (1200 reads + 700 writes) * 4 HBAs * 32KB = 243,200KB/s

If we know two of the values we can compute the third:

- A workload of 40 requests per second and 800KB/s throughput:
  - 800KB/s / 40 IOPS = 20KB average IO size

Note that we do not know that all of the IOs are 20KB, only that the average is 20KB per IO for that measurement. In most cases, the average is good enough for sizing a solution.

However, if we do not know at least two of the three values we cannot accurately describe the basic parameters of the workload. For example:

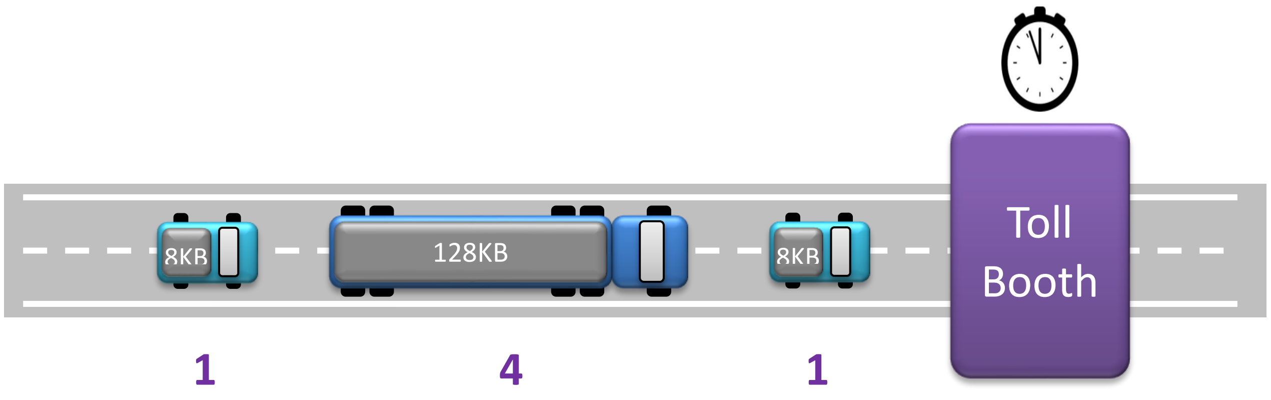
- My application requires 15,000 IOPS
  - What is the IO size?
  - What is the throughput?

Response Time

The time required to send, process, and receive acknowledgement (ACK) for a request is known as Response Time (RT). A read request involves sending a request for data, processing time to retrieve the data, and returning the data payload. A write request involves sending a data payload, processing time to store the data, and returning confirmation (ACK) that the write has completed.

The size of a storage IO request impacts how long it takes to process and, in many cases, how much work is required to process the request. The larger the request (IO Size), the longer it will take to process – which increases response time.

Consider our toll road example. A small car can pass through the toll booth in less time than a large truck. Response time is analogous to the vehicle entering, paying the toll, and then exiting the toll both.



In the example above, the response time for a small IO request is 1 millisecond (1ms) and response time for a large IO request is 4ms.

- The average response time is 1ms + 4ms + 1ms / 3 = 2ms (avg. RT)

Note: if provided an average response time we cannot tell if all of the response times are the same. There may be peaks or "spikes" hidden in the average.

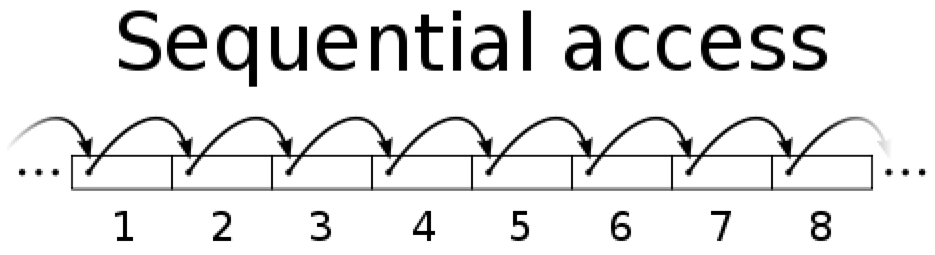
Read/Write Ratio

How many requests are reads compared to writes? A workload that is all reads may be expressed as 100% Reads, or 0% Writes. Another workload that has three write requests for every read request may be expressed as 75% Writes, 25% Reads, or a 1:3 Read/Write Ratio.

The read/write ratio is important because in most cases there is less effort required to process reads than to process writes. Thus, the more writes there are, the more work is done to process the writes. This is especially true for storage systems where a write request can require from 1.1x to 6x the amount of work that the same size read request may require.

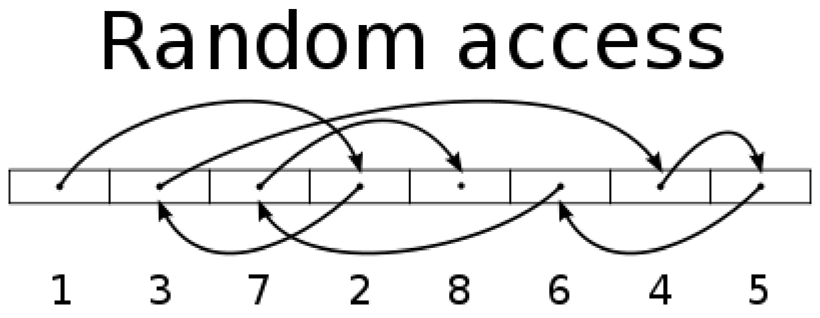
Random vs. Sequential

Sequential IO is a sequence of requests accessing data blocks that are adjacent to each other:



Reference: http://en.wikipedia.org/wiki/File:Random_vs_sequential_access.svg

Random IO is the opposite, a sequence of requests that are not adjacent to each other and may not have any regular pattern:

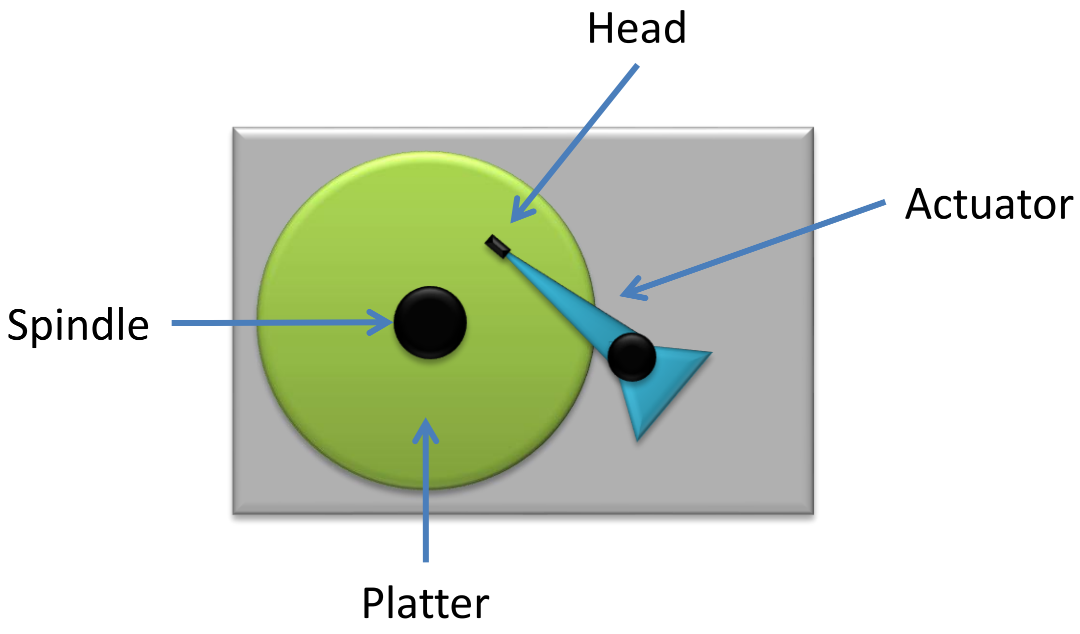


Linear Media

Tape devices such as LTO, DAT, DDS, and even old 9-track reel-to-reel systems can store data in relatively compact space, typically at lower cost than hard drives or memory devices. However, access to a specific piece of data requires loading the correct tape and then scanning or seeking to a specific location on the tape which can take seconds or even minutes to perform. Because of the nature of the linear storage format, tape drives are best used for sequential IO patterns such as backups and restores. However, tape media tends to break down over time, and the combination of reliability concerns and slower performance have led to hard drive-based replacements for most backup and archive applications.

Rotating Media

Traditional hard drives contain one or more platters that rotate at high speeds, and one or more heads that perform reads and writes to a magnetic media on the surface of the platters. An actuator arm moves the head in order to reach different regions on the platter.



Each move of the head to a new position – known as “seek time” – takes time away from reading or writing data. For this reason, rotating media drives perform best with sequential IO where the data is adjacent and lined up requiring little to no head movement.

Traditional hard drives also tend to work very well with larger IO sizes. The best sequential access performance for a rotational hard drive typically occurs with IO sizes of 256KB to 512KB, and sometimes as large as 4MB for video applications.

Random Access Media (RAM)

In contrast, most forms of memory – DRAM, SRAM, NVRAM, and Flash – process random requests much easier because there is no head that needs to be moved before IO can be processed. In other words, memory devices do not incur seek as do rotational devices.

Memory devices also process IO requests much faster than hard drives or tape devices. The memory devices have lower response times and higher throughput, but they also cost much more on a $/GB basis than hard drives or tape devices.

Response Times by Media Type

Recall that Response Time (RT) measures the full round trip time required to send, process, and acknowledge an IO request. The type of media being accessed has a significant effect on the response time, as does the IO size.

Most digital media – including RAM, Flash, and Hard Drives – measure response time performance for small block, random IO requests, typically 4KB or 8KB in size. The response time comparison between these device types is dramatic – sometimes one or two orders of magnitude difference:

Device Type	Average Response Time (ms)
Server SDRAM	Less than 0.001
PCIe Flash Card	0.02 – 0.10
Flash Drive	0.10 – 1.0
15K RPM HD	3 – 4
10K RPM HD	5 – 6
7.2K RPM HD	10 – 12

- Note: average response time for 4KB to 8KB request

You Say Potato, I say Pitato…

Before we go further, we need to address two frequent points of confusion regarding units of measure: bits vs. bytes, and decimal vs. binary.

Bits vs. Bytes

Bits are referenced with a lower-case ‘b’ and Bytes are referenced by an upper-case ‘B’.

Eight bits equals one byte: 8b = 1B

FUN FACT

- 4 bits is called a "nybble" (aka "nibble" - a morsel or small bite)

File sizes and media capacity are measured in Bytes or multiples.

The storage and networking disciplines use both bits per second and bytes per second measures. Most interface metrics report Throughput as Bits per Second (lower case "b"/s):

- 10Gb/s Ethernet Network Interface Card (NIC)
- 16Gb/s Fibre Channel Host Bus Adapter (HBA)
- 40Gb/s Infiniband Host Channel Adapter (HCA)

Most capacity metrics refer to Throughput as Bytes per second (upper case "B"/s):

- 120MB/s – uncompressed speed of LTO-4 tape drive
- 477MB/s – 4K video playback with RGB 4:4:4 codec
- 1TB/hr – backup speed for large data warehouse

We can express Throughput in either form. For example:

- A modern 16Gb/s Fibre Channel Host Bus Adapter (HBA) is rated for 16Gb/s
    - 16Gb/s ÷ 8b/B = 2GB/s
- The HBA can process 16Gb/s or 2GB/s

While individual network interfaces typically measure and report throughput in bits per second, it is very common to refer to the total workload in bytes per second – probably because I/O Size (in Bytes) times I/Os per second = Throughput in Bytes per second.

This distinction is important because storage and networking teams frequently need to work together and sometimes do not realize they are talking about different units of measure. A network interface may provide 10Gb/s, but if the application requires 2GB/s (16Gb/s) the system will need at least two 10Gb/s connections.

Decimal vs. Binary

You just purchased a new computer with a 2TB hard drive, and we know that 2 Terabytes (TB) equals 2000 Gigabytes (GB) of capacity. But after installing the operating system the computer shows the hard drive has roughly 1,862GiB of usable capacity. Where did the rest of the space go? And what is that 'i' in the middle of GiB?

Ironically they are the same capacity – the difference is units of measure: TB vs. TiB, GB vs. GiB, etc.

Look closer at the packaging on a hard drive and you will see the drive manufacturer lists capacity measured as 1 GB = 1,000,000,000 Bytes. Thus, the 2TB SATA drive is 2,000,000,000,000 Bytes.

The storage industry uses a decimal (base10) system for raw capacity using multiples of 1000 (10^3):

Decimal						
1 kilobyte (KB)	=	10^3	=	1000^1	=	1000 bytes
1 metabyte (MB)	=	10^6	=	1000^2	=	1000 kilobytes (KB)
1 gigabyte (GB)	=	10^9	=	1000^3	=	1000 megabytes (MB)
1 terabyte (TB)	=	10^12	=	1000^4	=	1000 gigabytes (GB)
1 petabyte (PB)	=	10^15	=	1000^5	=	1000 terabytes (TB)
1 exabyte (EB)	=	10^18	=	1000^6	=	1000 petabytes (PB)

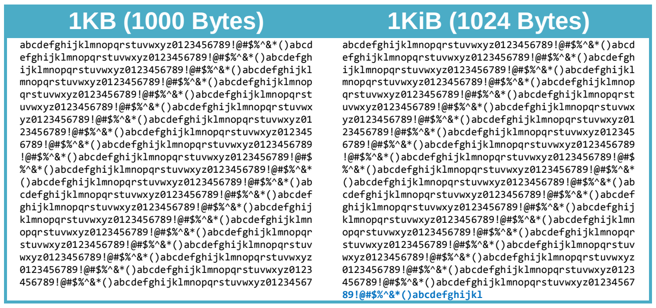
However, computer systems use binary (base2), and the units of measure are based on powers of 1024 (2^10):

Binary						
1 kibibyte (KiB)	=	2^10	=	1024^1	=	1000 bytes
1 mebibyte (MiB)	=	2^20	=	1024^2	=	1000 kibibytes (KiB)
1 gibibyte (GiB)	=	2^20	=	1024^3	=	1000 mebibytes (MiB)
1 tebibyte (TiB)	=	2^20	=	1024^4	=	1000 gibibytes (GiB)
1 pebibyte (PiB)	=	2^20	=	1024^5	=	1000 tebibytes (TiB)
1 exbibyte (EiB)	=	2^20	=	1024^6	=	1000 pebibytes (PiB)

Convert between decimal and binary as follows:

- 1TB = 1,000GB = 1000^4 = 1,000 * 1,000 * 1,000 * 1,000 Bytes
- 1TiB = 1,024GiB = 1,024^4 = 1,024 * 1,024 * 1,024 * 1,024 Bytes
- 1000^4 / 1024^4 = 1,000,000,000,000 / 1,099,511,627,776 = 0.909 TiB

Below we can see the difference between 1 KB (1000 Bytes) and 1 KiB (1024 Bytes) from the same file containing ASCII data:



1KB (1000 Bytes)	1KiB (1024 Bytes)
abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcd	abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcd
efghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefgh	efghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefgh
ijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijkl	ijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijkl
mnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnop	mnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnop
qrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrst	qrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrst
uvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwx	uvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwx
yz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz01	yz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz01
23456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz012345	23456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz012345
6789!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123456789	6789!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123456789
!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123456789!@#$	!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123456789!@#$
%^&*()abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*	%^&*()abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*
()abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()ab	()abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()ab
cdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdef	cdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdef
ghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghij	ghijklmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghij
klmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmn	klmnopqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmn
opqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqr	opqrstuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqr
stuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuv	stuvwxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuv
wxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz	wxyz0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz
0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123	0123456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz0123
456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz01234567	456789!@#$%^&*()abcdefghijklmnopqrstuvwxyz01234567
	89!@#$%^&*()abcdefghijkl

Those last 24 characters in **blue** are the 2.4% difference between 1 KB and 1 KiB. And because these measures are powers of 1000 and 1024 respectively, as the unit of measure increases so does the disparity between then increase.

In other words, 1 KB vs. 1 KiB is a ~2.4% difference, while 1 MB vs. 1 MiB is a ~4.63% difference – it is a compounding effect.

The discrepancy grows as the prefixes increase. Below is the table showing this effect:

Prefix	Decimal	Binary	Dec to Bin
Kilo	1,000	1,024	97.656%
Mega	1,000,000	1,048,576	95.367%
Giga	1,000,000,000	1,073,741,824	93.132%
Tera	1,000,000,000,000	1,099,511,627,776	90.949%
Peta	1,000,000,000,000,000	1,125,899,906,842,624	88.818%
Exa	1,000,000,000,000,000,000	1,152,921,504,606,846,976	86.736%

Reference: http://en.wikipedia.org/wiki/Kibibyte

Workload Profiles

What are Workload Profiles?

In the most generic sense, a workload profile describes the behavior of an application, or portion of an application with respect to how it performs work.

A storage workload profile describes several aspects of how an application accesses data on a storage device – such as a hard drive, USB stick, memory (e.g. DRAM or NVRAM), or external storage array (e.g. SAN, NAS, etc.).

The characteristics of a workload profile affect how much effort is required to perform the work. Typically, less effort required means less equipment and lower cost to provide the desired performance. Conversely, more effort typically means more equipment and higher cost.

Workload profiles for storage systems are typically described and grouped by key “Workload Characteristics” – i.e. attributes they have in common:

- Are the IO requests large or small?
- Are the IO requests reads, writes, or mixed reads and writes?
- Is the access pattern sequential or random?

Often, applications have multiple workload profiles. For example, an order processing system involves several operations:

- Order Entry – primarily data input (**writes**) of short line items (**small**), often many unrelated orders being processed at the same time (**random**)
- Packing and Shipping – data input and output to read order details, update inventory and shipping records (**mixed read/write**)
- Reporting and Data Backups – primarily data output (**reads**) of grouped information (**large**), often using indexes or tables scans (**sequential**)

Workloads and Performance

How does a workload profile affect performance?

Consider the following guidelines:

1. Large requests take longer to process than small requests.	• The larger the IO size, the more time required to send and process the request.
2. In most situations, writes require more work than reads.	• The more writes that a workload performs, the more work a storage system needs to do. • Most media types have higher response times for writes and often have additional overhead due to data protection, such as RAID.
3. Some storage technologies perform better with random IO patterns. Others perform better with sequential IO patterns.	• Traditional hard drives generally perform best with sequential IO patterns by avoiding moving the drive head (seek) to a new location. • Memory and Flash devices tend to perform better with random IO patterns because there is no head movement and IOs can be parallelized to multiple cells.

Workload Variations

Sustained Workload

Activity or a level of performance that is maintained for several minutes or hours is known as a sustained workload. The importance of the sustained workload is that it is the best indicator of activity that lasts for a meaningful period of time and accurately represents a level of performance that needs to be supported by infrastructure.

When sizing for performance we typically focus on Sustained Peak. This represents a performance level that must be maintained for several minutes or a few hours without the extreme highs from bursts or lows of inactivity that can be hidden by averages.

The key focus of performance sizing is to ensure that a system can support sustained workload peaks without significant performance degradation.

Averages

A workload average typically represents the arithmetic mean for a set of values over a known time period. The average gives us an idea of the general behavior over time, but it does not provide any detail about the lows and highs (peaks) in a time series. An average can mask very low or no activity that may be planned – such as maintenance work. The average can also hide very high activity or short bursts. By definition, averages change based on the time period that is examined.

A big problem with averages is that picking the wrong time period – too short, too long, or wrong start and stop times – can result in an average that is far different from the important period for applications and business activity. Often, it is not known when the important activity is occurring vs. activity that is not important.

The main risk of sizing to a workload average is that the storage infrastructure may not be able to deliver adequate performance during workload peaks.

Peaks and Bursts

A workload peak represents the highest level of performance achieved by an application, device, or infrastructure during a period of time. There can be separate peaks for different workload characteristics. For example:

- Peak IOPS during month end order processing
- Peak Throughput (MB/s) during database backups
- Peak Response Time during nightly batch processing

Very short peaks are sometimes referred to as bursts, typically lasting less than one second or as long as a couple of minutes. In storage and network environments it is common to observe very short bursts of activity that can be several times the sustained peak workload, and may be orders of magnitude higher than the average workload.

The primary question to consider is: How long does the burst of activity last?

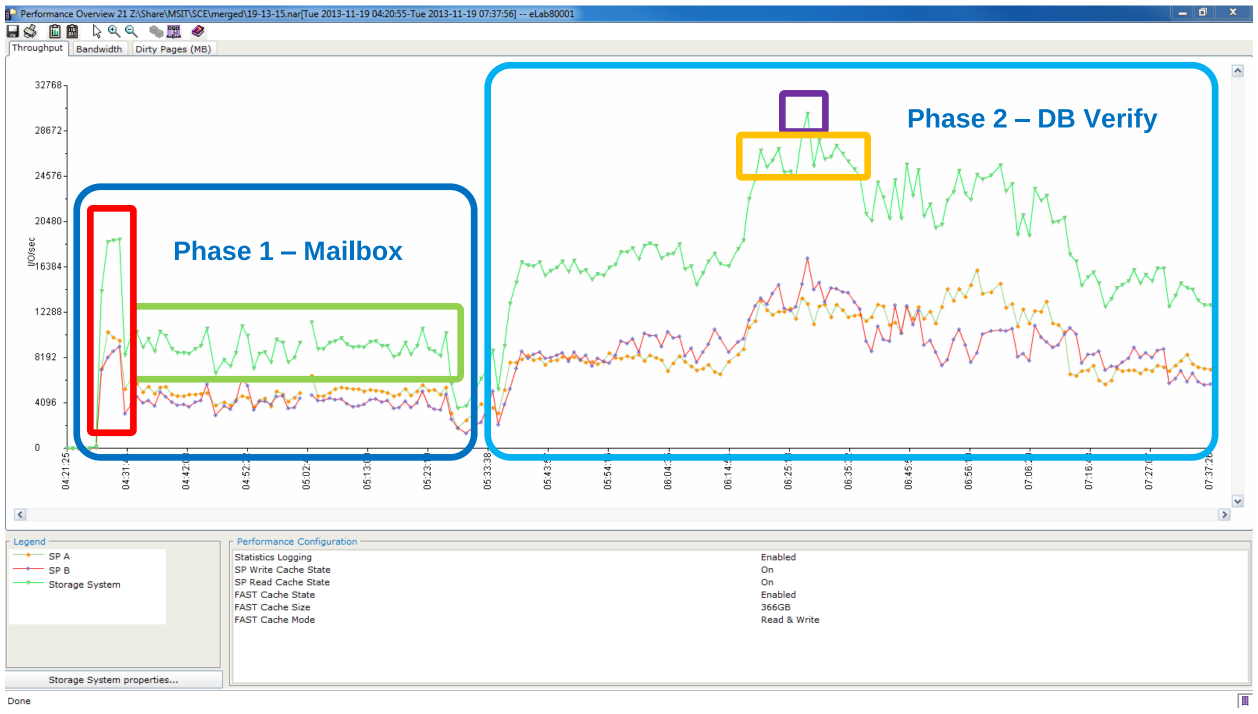
If a burst is very short, it may be a distracter or anomaly that can be ignored for sizing purposes. The risk of sizing to very short bursts is that they do not accurately represent the required level of performance and can lead to a solution that may be too costly or too complex to operate and maintain.

In most cases, a solution should be sized to support the sustained peak workload which ensures that performance is acceptable during the busiest sustained period of work.

Identifying Workloads Variations

The example below is a JetStress test that simulates a Microsoft Exchange workload. The graph comes from Unisphere® Analyzer showing total IOPS for a VNX® array that is providing SAN storage for the test environment. The JetStress test runs in two phases:

4. **Mailbox Tests** – simulating messages being sent and received. This portion performs a mix of random reads and writes and best approximates performance that users will experience.
5. **Database Verify** – validates the integrity of the Exchange DB files. This simulates a batch process that typically runs at night or between time zones and has less impact on users.



Performance Overview 21 Z:\Share\MSIT\SCE\merged\19-13-15.nar[Tue 2013-11-19 04:20:55-Tue 2013-11-19 07:37:56] -- eLab80001

Performance Configuration

Statistics Logging	Enabled
SP Write Cache State	On
SP Read Cache State	On
FAST Cache State	Enabled
FAST Cache Size	366GB
FAST Cache Mode	Read & Write

- During phase 1 of the test there is an initial peak over 18,750 IOPS that lasts for three minutes – outlined in **red**.
- The remainder of phase 1 – outlined in **green** – shows an average of approximately 9,500 IOPS for nearly one hour, with short bursts and lows during that hour.
    - The short peak at the beginning of the test is due to all of the JetStress threads starting and is not normal workload.
- During phase 2 of the test we can see a short burst – outlined in **purple** – to over 30,000 IOPS, but it occurs only once during a sustained peak of ~26,000 IOPS – outlined in **orange** – that lasts for approximately fifteen minutes.

    - In most scenarios it is acceptable to ignore the burst of 30,000 IOPS, and instead focus on sizing the solution to support ~26,000 IOPS during the DB Verify portion of the test.

Typically, the storage solution will be configured in order to satisfy the average of 9,500 IOPS. In many scenarios, sizing of the storage will include a growth factor of 10%, 20%, or possibly more to account for business growth over time.

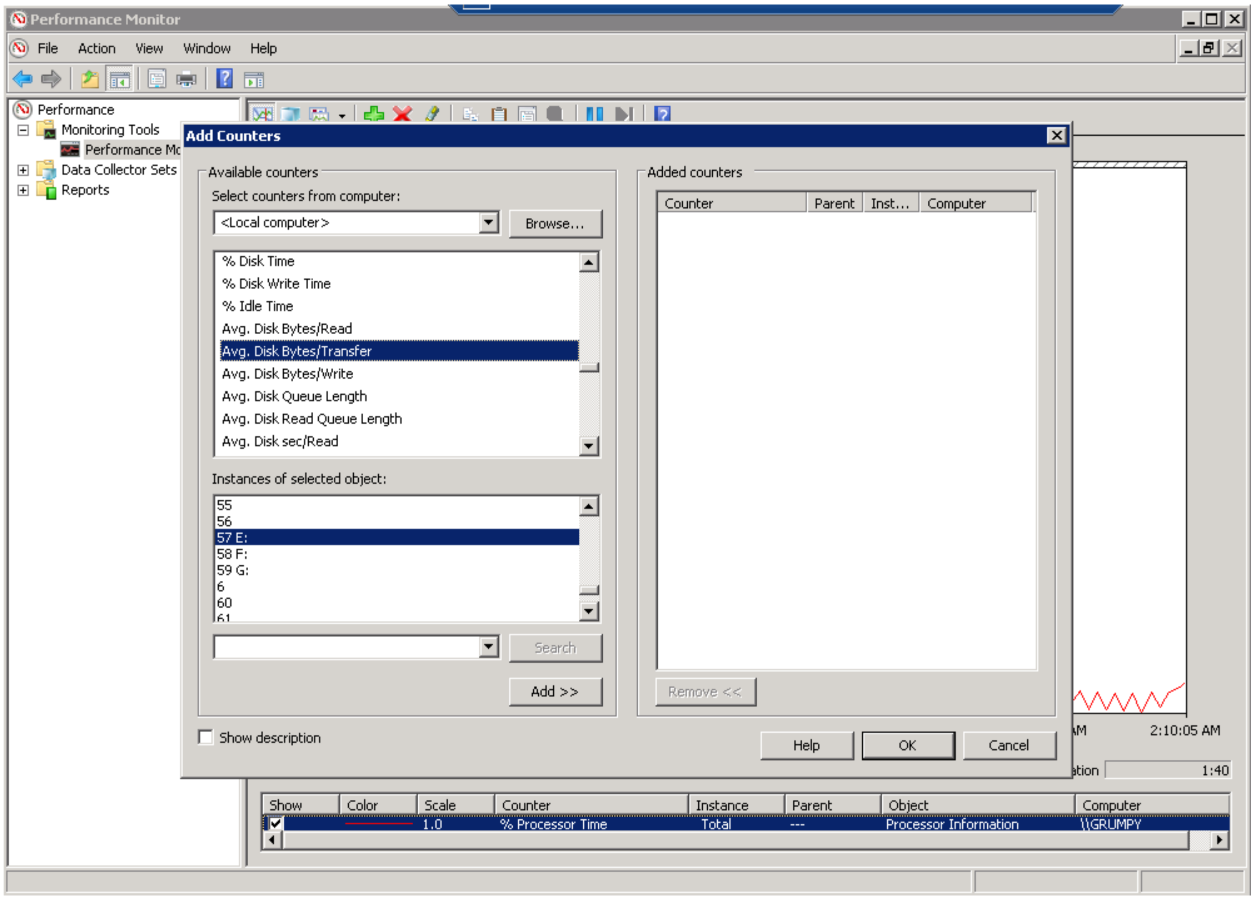
Identifying Workload Profiles

Windows Perfmon utility

Perfmon is a utility included in most versions of Microsoft Windows operating system. It can report on a variety of metrics for CPU, memory, network, disk, and application resources.

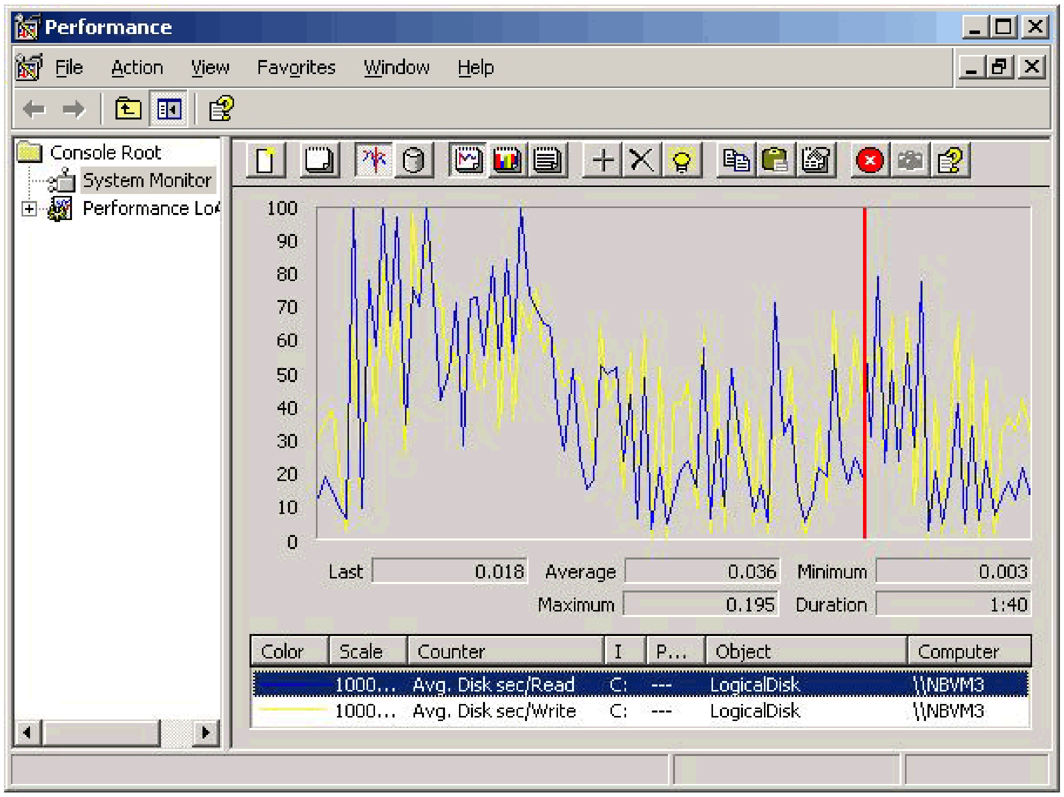
For storage environments, there are several counters available for individual drives, or aggregate counters for all drives connected to a server. The image below shows an example of some key metrics in the Physical Disk category:

- IO Size = Disk Bytes/Transfer (also: Bytes/Read, Bytes/Write)
- IOPS = Disk Transfers/sec (also: Reads/sec, Writes/sec)
- Response Time = Avg. Disk sec/Transfer (also: sec/Read, sec/Write)



Show	Color	Scale	Counter	Instance	Parent	Object	Computer
✓	—	1.0	% Processor Time	Total	---	Processor Information	\\GRUMPY

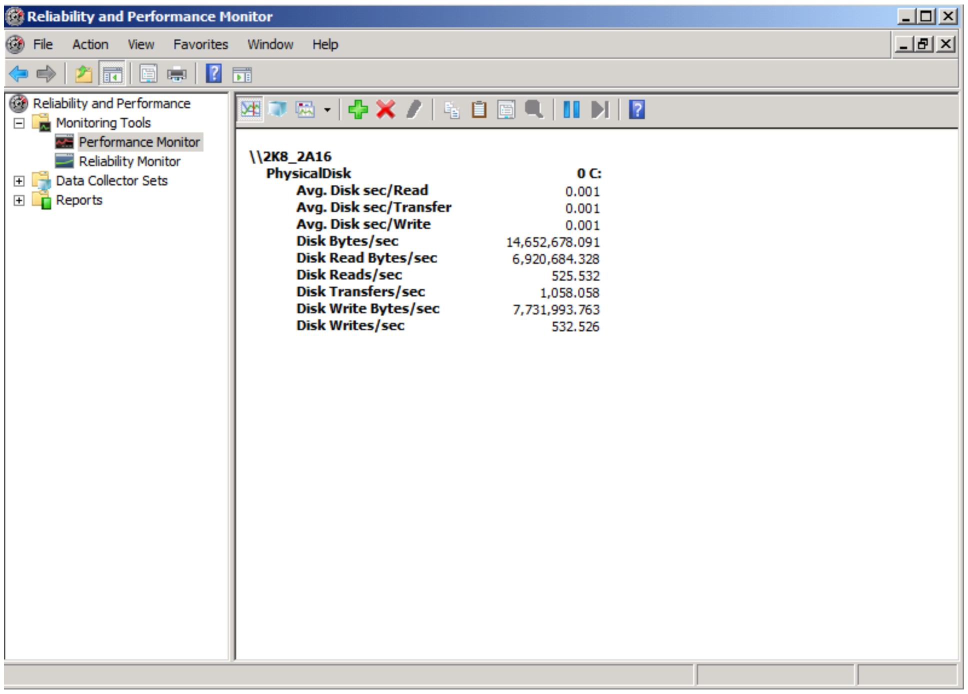
Perfmon metrics can be graphed over a period of time:



Last 0.018 Average 0.036 Minimum 0.003
Maximum 0.195 Duration 1:40

Color	Scale	Counter	I	P...	Object	Computer
	1000...	Avg. Disk sec/Read	C:	---	LogicalDisk	\\NBVM3
	1000...	Avg. Disk sec/Write	C:	---	LogicalDisk	\\NBVM3

Or show current values in text format:


Reliability and Performance Monitor
File Action View Favorites Window Help
Reliability and Performance
Monitoring Tools
Performance Monitor
Reliability Monitor
Data Collector Sets
Reports

```
\\2K8_2A16
  PhysicalDisk                              0 C:
      Avg. Disk sec/Read                   0.001
      Avg. Disk sec/Transfer               0.001
      Avg. Disk sec/Write                  0.001
      Disk Bytes/sec              14,652,678.091
      Disk Read Bytes/sec          6,920,684.328
      Disk Reads/sec                     525.532
      Disk Transfers/sec               1,058.058
      Disk Write Bytes/sec         7,731,993.763
      Disk Writes/sec                    532.526
```

Linux / Unix ‘iostat’ command

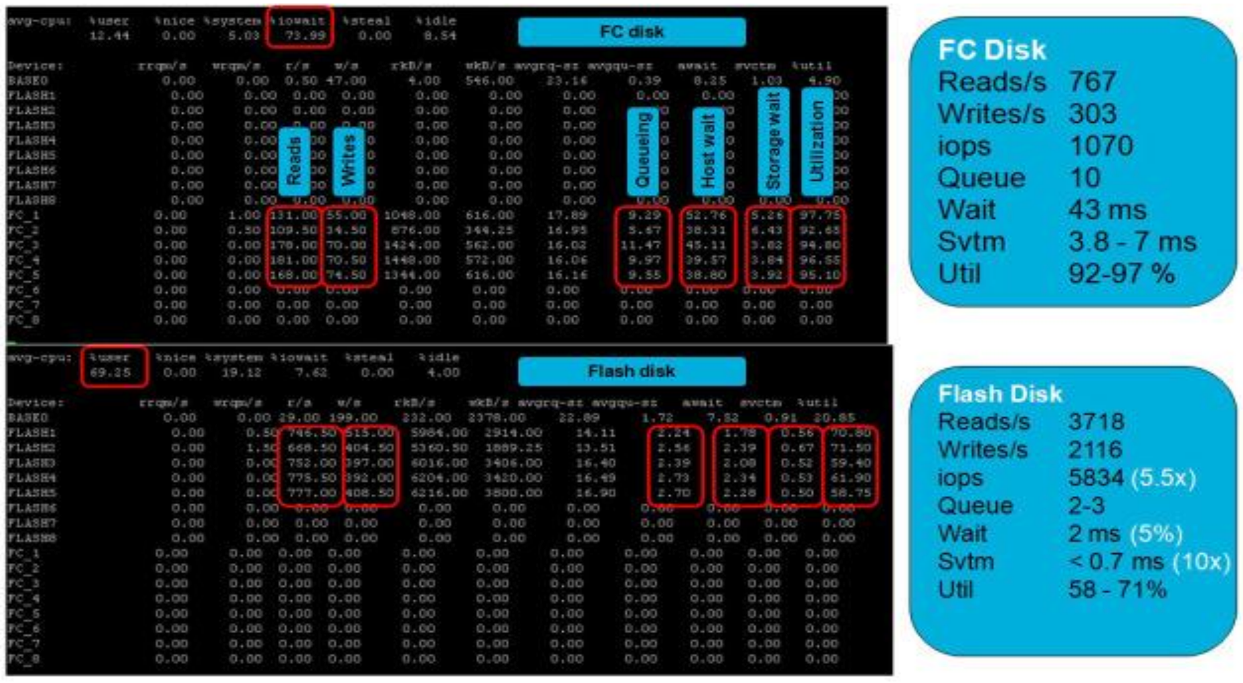
The ***iostat*** command available in several Linux and Unix distributions provides performance metrics for CPU, memory, network, and storage. The utility is invoked via command line with an interval in seconds, and optionally the number of intervals to report. There are also optional flags to format the metrics in easier to read units such as KB or MB. For example, the following will run iostat every 10 seconds for 5 minutes:

```
iostat -xk 10 5
```

The output format varies from version to version, but they generally report the following metrics – averages over the time interval specified in the command:

- Read IOPS = r/s, and Write IOPS = w/s
- Read Throughput = rkB/s, and Write Throughput = wkB/s
- IO Size = avgrq-sz
- Response Time = await

The example below shows performance comparison of Fibre Channel (FC) drives vs. Flash drives in a storage array – as observed from the Linux host.



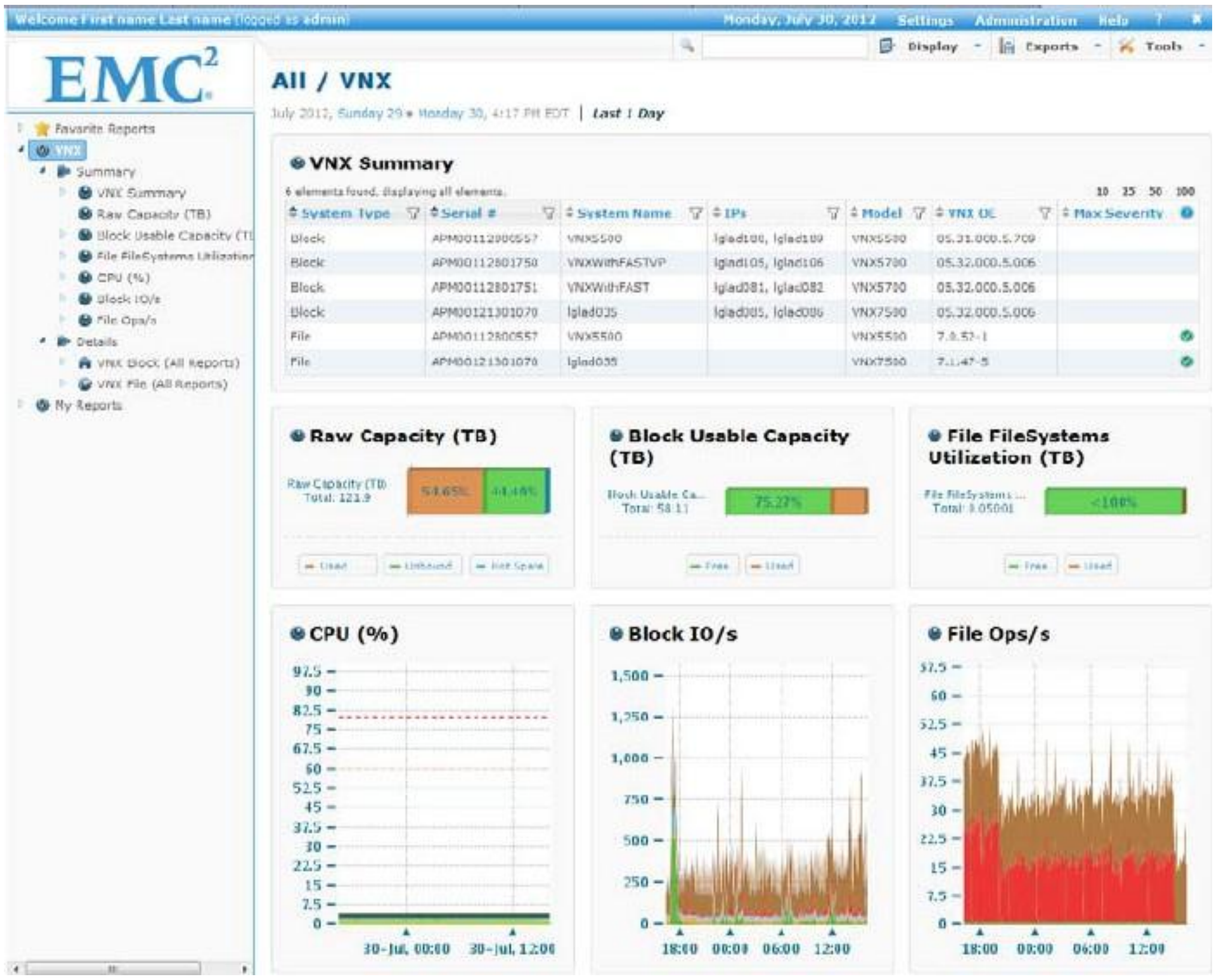
```
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          12.44    0.00    5.03   73.99    0.00    8.54

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await  svctm  %util
BASE0             0.00     0.00    0.50   47.00     4.00   546.00    23.16     0.39    8.25   1.03   4.90
FLASH1            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH2            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH3            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH4            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH5            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH6            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH7            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH8            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_1              0.00     1.00  131.00   55.00  1048.00   616.00    17.89     9.29   52.76   5.26  97.75
FC_2              0.00     0.50  109.50   34.50   876.00   344.25    16.95     5.67   38.31   6.43  92.65
FC_3              0.00     0.00  178.00   70.00  1424.00   562.00    16.02    11.47   45.11   3.82  94.80
FC_4              0.00     0.00  181.00   70.50  1448.00   572.00    16.06     9.97   39.57   3.84  96.55
FC_5              0.00     0.00  168.00   74.50  1344.00   616.00    16.16     9.55   38.80   3.92  95.10
FC_6              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_7              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_8              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
          69.25    0.00   19.12    7.62    0.00    4.00

Device:         rrqm/s   wrqm/s     r/s     w/s    rkB/s    wkB/s avgrq-sz avgqu-sz   await  svctm  %util
BASE0             0.00     0.00   29.00  199.00   232.00  2378.00    22.89     1.72    7.52   0.91  20.85
FLASH1            0.00     0.50  746.50  515.00  5984.00  2914.00    14.11     2.24    1.78   0.56  70.80
FLASH2            0.00     1.50  668.50  404.50  5360.50  1889.25    13.51     2.56    2.39   0.67  71.50
FLASH3            0.00     0.00  752.00  397.00  6016.00  3406.00    16.40     2.39    2.08   0.52  59.40
FLASH4            0.00     0.00  775.50  392.00  6204.00  3420.00    16.49     2.73    2.34   0.53  61.90
FLASH5            0.00     0.00  777.00  408.50  6216.00  3800.00    16.90     2.70    2.28   0.50  58.75
FLASH6            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH7            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FLASH8            0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_1              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_2              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_3              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_4              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_5              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_6              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_7              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
FC_8              0.00     0.00    0.00    0.00     0.00     0.00     0.00     0.00    0.00   0.00   0.00
```

Reference: http://bartsjerps.wordpress.com/2011/03/04/io-bottleneck-linux/

VNX Monitoring and Reporting

Monitoring and Reporting (M&R) for VNX arrays is a lightweight version of EMC Storage Resource Management (SRM) Suite. VNX M&R focuses on capacity and performance metrics for VNX arrays with historical trends and alerting for one to several VNX arrays.

Below is an example of the VNX summary reporting from Monitoring and Reporting package. Available metrics include IOPS, Throughput, Response Time, and several others for SAN and NAS functionality.



All / VNX

July 2012, Sunday 29 • Monday 30, 4:17 PM EDT | Last 1 Day

VNX Summary

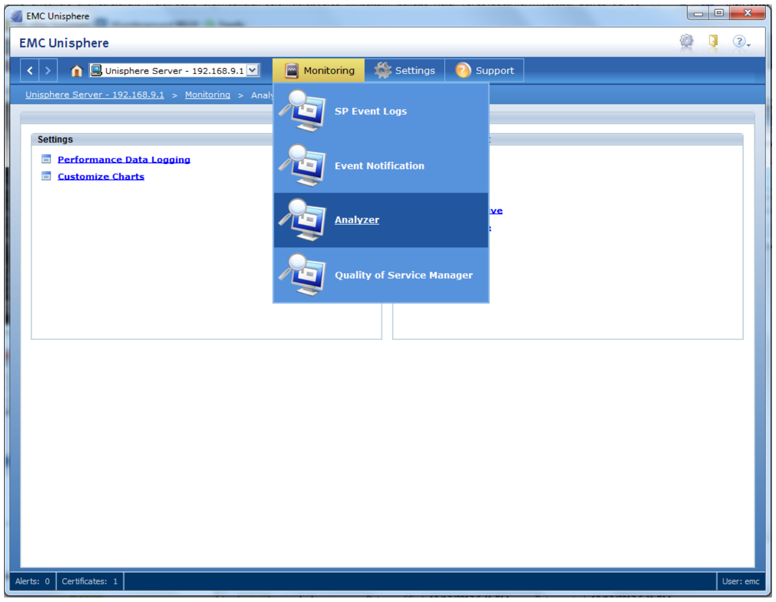
6 elements found, displaying all elements.

System Type	Serial #	System Name	IPs	Model	VNX OE	Max Severity
Block	APM00112900557	VNX5500	lglad100, lglad109	VNX5500	05.31.000.5.709	
Block	APM00112801750	VNXWithFASTVP	lglad105, lglad106	VNX5700	05.32.000.5.006	
Block	APM00112801751	VNXWithFAST	lglad081, lglad082	VNX5700	05.32.000.5.006	
Block	APM00121301070	lglad035	lglad085, lglad086	VNX7500	05.32.000.5.006	
File	APM00112800557	VNX5500		VNX5500	7.0.52-1	●
File	APM00121301070	lglad035		VNX7500	7.1.47-5	●

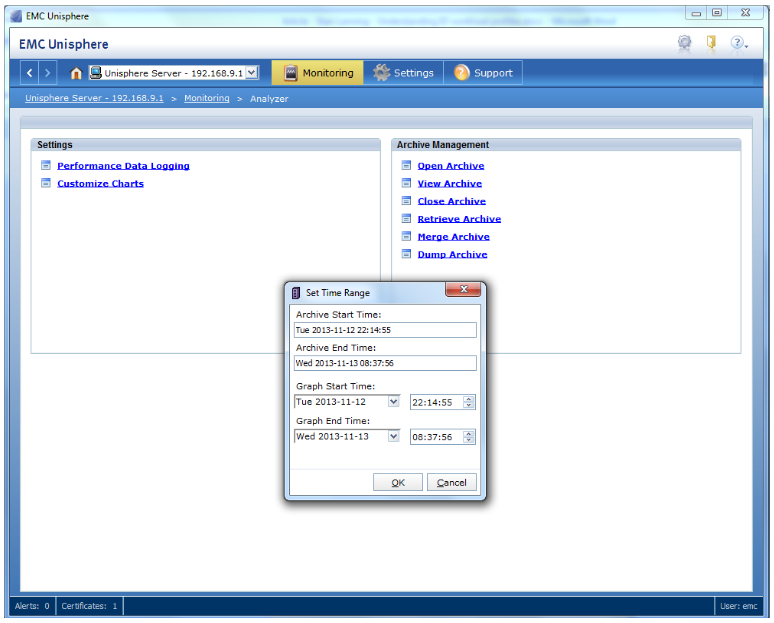
Raw Capacity (TB)
Raw Capacity (TB) Total: 121.9
54.65%
44.48%
Used
Unbound
Hot Spare
Block Usable Capacity (TB)
Block Usable Ca... Total: 58.11
75.27%
Free
Used
File FileSystems Utilization (TB)
File FileSystems ... Total: 3.05001
<100%
Free
Used
CPU (%)
30-Jul, 00:00
30-Jul, 12:00
Block IO/s
File Ops/s
18:00
00:00
06:00
12:00

Unisphere for VNX – Analyzer

Unisphere Analyzer is a performance reporting feature for EMC VNX arrays. It is launched from within the Unisphere for VNX Graphical User Interface (GUI) and can provide near real-time or historical views of a wide variety of array performance metrics. Access Unisphere Analyzer via the Monitoring -> Analyzer menu option.

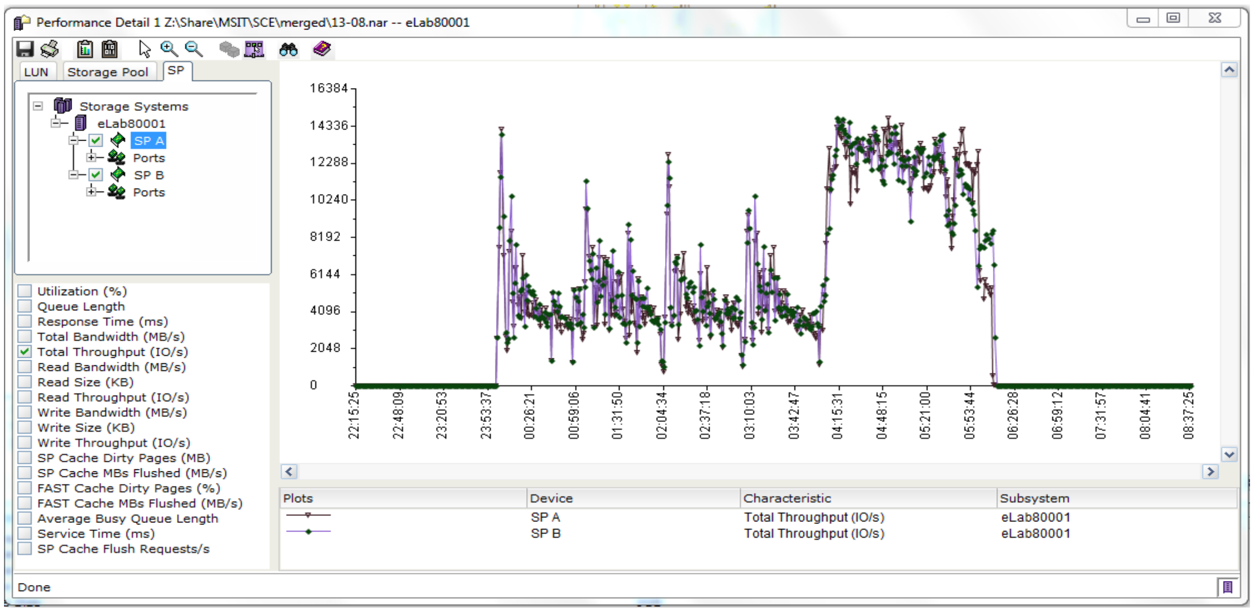


Historical metrics are collected in NAR (compressed archives) or NAZ (encrypted archives) files and later reviewed within Unisphere Analyzer. Here is an example opening a NAR file – a window pops up to show the date and time frame of the metrics within the file.



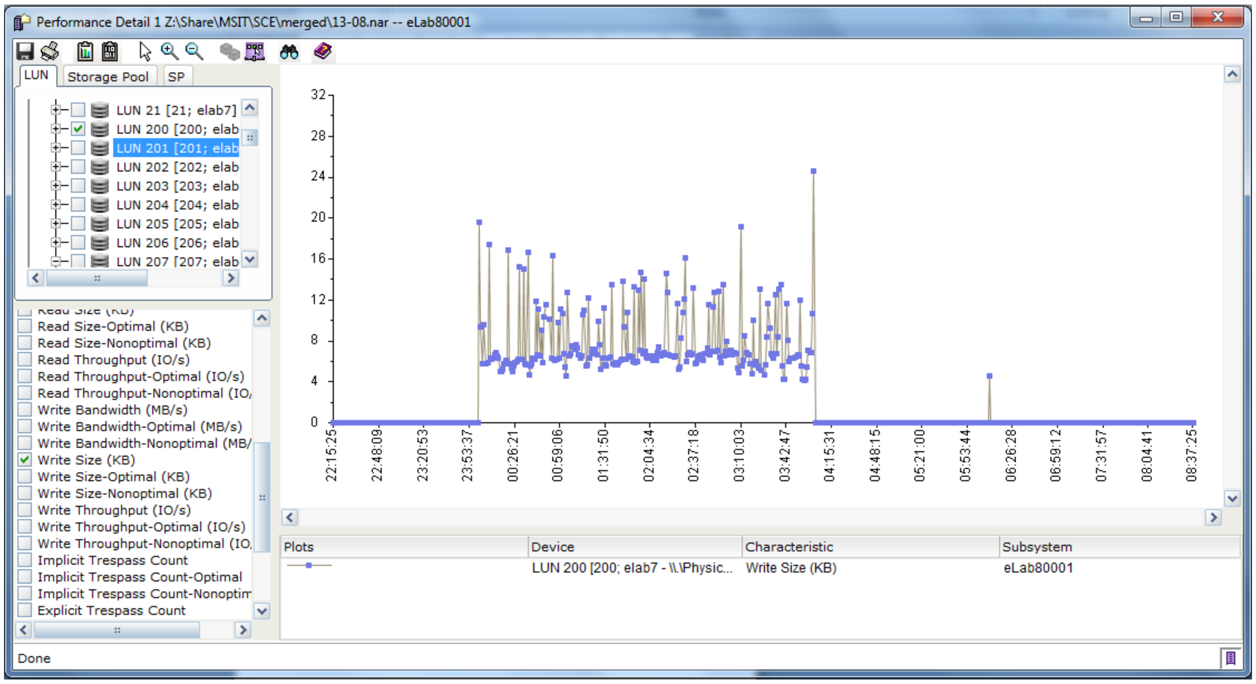
Archive Start Time: Tue 2013-11-12 22:14:55
Archive End Time: Wed 2013-11-13 08:37:56
Graph Start Time: Tue 2013-11-12 22:14:55
Graph End Time: Wed 2013-11-13 08:37:56

Several views are available including Performance Overview, Performance Summary, and Performance Detail. Below is the Performance Detail view showing total IOPS for each of the Storage Processors (SPs) on a VNX array.



Plots	Device	Characteristic	Subsystem
▼	SP A	Total Throughput (IO/s)	eLab80001
◆	SP B	Total Throughput (IO/s)	eLab80001

Unisphere Analyzer can drill down to individual LUNs and disks in the array. Here is an example of write IO size during a 4-hour application test – average is 8KB with short bursts as high as 24KB.

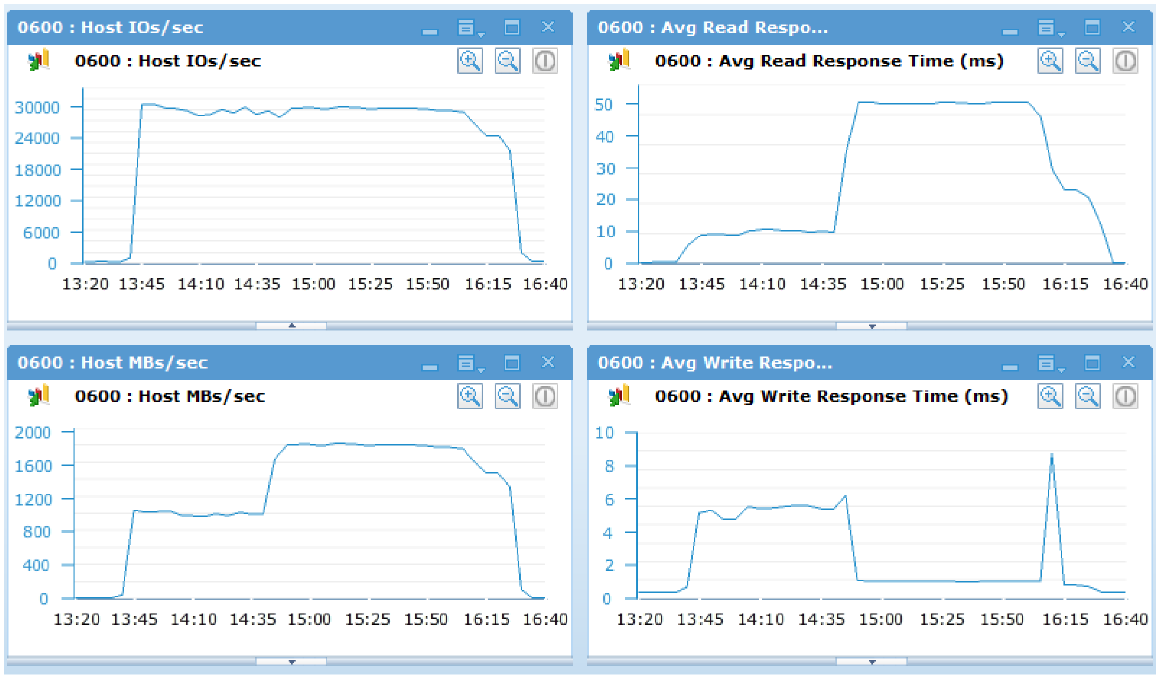


Plots	Device	Characteristic	Subsystem
■	LUN 200 [200; elab7 - \\.\Physic...	Write Size (KB)	eLab80001

Unisphere for VMAX – Performance Monitor

Unisphere for VMAX includes a performance monitoring section that can view near real-time and historical performance metrics for one or more Symmetrix VMAX arrays. The graphs below show several key metrics during a test workload preparing an array to run Microsoft Exchange.

- IOPS = Host IOs/sec
- Throughput = Host MBs/sec
- Response Time = Avg Read Response Time (ms), and Avg Write Response Time (ms)



Conclusion

It is important to understand the key characteristics of an application or process – the workload profile – which includes IO size, IOPS, Throughput, Response Time, Read/Write Ratio, and several other metrics. By understanding the workload profiles on a host or storage system we can ensure that solutions are correctly sized in order to optimize acquisition and operating costs for computing infrastructure.

For more information on performance data gathering and reporting reference:

- EMC VNX Monitoring and Reporting Demonstration Video
  - http://www.youtube.com/watch?v=SgqOtZcqn-A
- EMC Unisphere Analyzer for VNX Demonstration Video
  - http://www.youtube.com/watch?v=yCMZ_N7-p7A
- EMC Unisphere for VMAX Product Guide
  - https://support.emc.com/docu46997_Unisphere-for-VMAX-1.6-Product-Guide.pdf?language=en_US