The Ins and Outs of IOPS

Strengths and weaknesses of using IOPS as a measure of storage performance

WHITE PAPER BY BRAD BONN

Introduction

Recently, and over the last couple years, I've seen IOPS become a buzzword everywhere. Phrases such as “We have a one million IOPS-capable SAN so storage shouldn't be having a problem,” or “I really need IOPS visibility” are popping up in conversations and appearing in online communities like the weeds in my garden. It makes sense that we're seeing the topic appear, because there is a decided lack of measurement standards for the overall performance of an end-to-end storage solution. Earlier on in my years of IT, disk I/O capacity has classically been “measured” in terms of the number of acronyms you could rattle off. “Yeah, I've got a 12-spindle RAID 0+1 of 15K 146GB SAS drives in each of my shelves.” That's all well and good, but what does it translate to in terms of actual usability? How many databases can it support, and of what level of transactional utilization? How many users on an Exchange system could it handle? What kind of file server load could it deliver? The universal answer is “it depends.” Application and file system diversity, the sharing of storage hardware through SAN/NAS devices, and the additional levels of sharing added through virtualization make unexpected results...well, expected.

The market abhors a vacuum, and when there is a clear need, vendors and integrators alike will move to try and fill it. The need in this case is the simplification of storage utilization both in terms of need from the application point of view, and in terms of delivery from the storage vendor point of view. Voila, IOPS.

It's a very simple concept, which is part of the reason it's become so widely used. “I/O Operations Per Second” is an easily understood and communicated unit of measurement. Unfortunately, it's also very easy to over-simplify. IOPS (or IOps or IOPs depending on the phrase you’re actually abbreviating) only describes the number of times an application, OS, or VM is reading and/or writing to storage each second. This sounds like a useful metric, because it is! More IOPS means more disk I/O, and if all IOPS are created equal we can measure disk activity with it alone. But the problem is, they aren't.

This topic is hotly debated on all sides, and having spoken with storage vendors, IT admins, and SMEs, I’ve come to the conclusion that as important as IOPS are, they aren’t the only metric you need to examine when you measure storage performance.

The goal of this document is primarily to outline the strengths and weaknesses of the use of IOPS in measuring storage capabilities; specifically from the perspective of shared storage. Along the way, we will cover some of the basic concepts surrounding shared storage itself and the implications that design choices in building a solution can have upon the performance and price of your infrastructure.

Shared Storage Fundamentals

If you're already familiar with shared storage technology in general, feel free to skip ahead to the next section, but it's helpful to review the components of a SAN to best understand the impact IOPS can have.

Where we keep the bits and bytes of data in our server farms has come a long way. Just spend some time on Wikipedia looking up things like "core memory" and punch cards to get a reminder of the evolution that storage has undergone over the years. Plus, not only has the medium by which we store data changed, but also the method by which we get information in and out of those sources has become just as diverse.

In the mainframe days, "shared storage" was a redundant title. All the computing resources for the building or company, including storage, were centrally located and therefore shared. Whatever tapes, disks or memory housed the data was all connected to the same core system, or systems. The resulting hierarchy was then logically very star-shaped with terminals connecting centrally in order to utilize the mainframe.

Modern computing is much more dense, and likewise, much more distributed. The giant mainframes of the past with their singular presences have been replaced by sprawling, interconnected datacenters consisting of dozens, hundreds, or even thousands of individual servers that all communicate internally and externally with other servers or personal computers. With each node being able to house its own storage devices, where data is located becomes equally as distributed.

These days, unless your "server room" consists of a few PCs and a SOHO router, you're probably using a SAN or NAS in your infrastructure to share data between servers. Storage solutions like these make the allocation and migration of logical disks more manageable and fluid, to the point where directly-attached storage, or DAS (local disks contained inside each individual server) is practically never used in the datacenter any longer. Small infrastructures can still benefit from the lower initial investment of DAS, but in a rapidly-growing environment, which call for business continuity from natural disasters, shared storage is critical to enabling rapid expansion, and vastly improving cost density.

Whether you are using a storage area network or network-attached storage system, the end goal is the same: having a one-to-many relationship between a storage device and the computers that access it. Making that relationship happen involves various pieces of physical equipment and layers of logical abstraction, and while this whitepaper makes no claim to be a definitive guide on the broad topic of shared storage, we need to discuss some of the complexities involved in order to get a clear idea of what IOPS really means for such a system.

Links in the chain

Shared storage devices, regardless of their make, model, size or configuration all consist of the same general components. Any single read or write command has to traverse at least part of this chain in order to reach its destination, whether it is the active memory of a server, or the bare metal of a spinning disk. Every single link in this chain will affect the speed at which the data reaches where it's going, as well as how many of those operations can be executed within a span of time.

The "weakest link" in this chain will determine the maximum number of IOPS and the total I/O bandwidth of the system. **Figure 1** gives a generalized but effective view of what levels exist within this "chain." Notice that at every step of the way, additional connections are possible. In a block-level shared storage system such as fibre channel or iSCSI, (generally what would be considered a SAN) a LUN (logical unit number) can be accessed by multiple hosts (the effective one-to-many use case itself), but then those hosts can talk to multiple storage devices containing multiple disks via multiple paths.

In a file-based shared storage system such as an NFS/CIFS filer (considered a type of NAS,) a LUN isn't used. Instead, filers will present network share locations, which can be accessed as logical disks by hosts and VMs. This option tends to be much more affordable since it can leverage existing networks and does not require as much specialized hardware. However this comes with a performance cost which may not always suit the needs of high-demand tier1 applications.

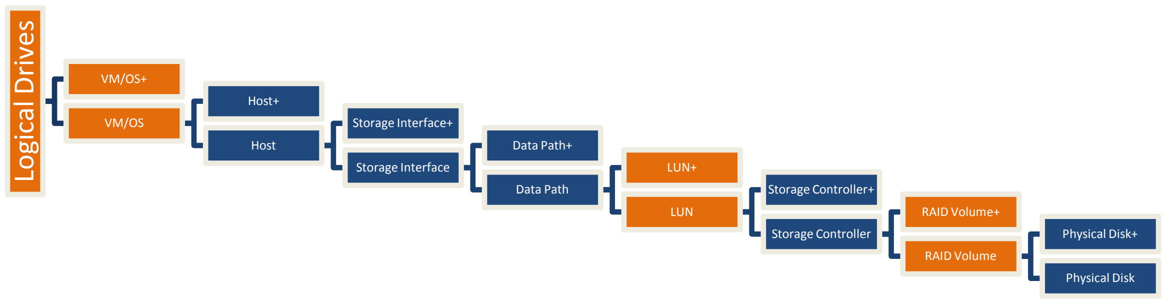


Figure 1: Abstraction of the Layers Beneath a Shared Storage Volume
Items in orange = Logical Abstraction Levels
Items in blue = Physical Devices

Starting at the "bottom," we find the physical disks themselves. The faster the drive and its interface, the more expensive it becomes. Choosing the underlying disk technology will heavily impact the SAN or NAS' total I/O capability and storage capacity. Lots can be done to get the most out of the spindles, but the buck stops here in the end.

A set of inexpensive magnetic SATA drives spinning at only 7200RPM bring a great cost to storage density ratio, but at the expense of limited speed. I/O-light applications or data archives are always well-suited to these. On the other hand, a set of 15,000RPM serial-attached SCSI (SAS) drives with hefty memory caches on-board, or solid-state drives (SSDs) with no moving parts built entirely from flash chips will bring astonishing speed to the table for hefty database operations or disk-intensive apps at the cost of serious hit to your budget. Larger SANs will contain a mixture of these kinds of drive technologies, allowing for the intelligent balancing of disk loads to match performance tiers.

On the next link in the chain, these physical disks are bundled into logical groups, often known as a RAID (Redundant Array of Independent Disks) which ensure the protection of the data against disk failure, and can potentially speed up bare disk operations through parallel reads and writes. However, depending on the type of RAID configuration in place, the total performance of the disks can be *negatively* impacted as well. For a detailed description of the types of RAID that are out there and their effects on performance, this article online can help:
http://www.accs.com/p_and_p/RAID/BasicRAID.html

Many modern storage systems take array configuration completely out of the hands of the storage administrator. This can greatly simplify the bare disk component of the storage and enforce best practices across tiers.

In order for the disks in an array to function as a logical unit, there must be a device or software to organize them into such a structure. In a shared storage model, the storage processor (*or storage controller*) handles this. Usually housed inside the chassis of the Storage Array or Filer, the storage processor is a self-contained computer system that handles all I/O for the device. It manages each I/O operation written to and read from the disks, as well as manages the communication to the hosts utilizing the shared storage via the various mediums available.

A storage array has multiple redundant and load-balancing storage processors, and can be made up of multiple physical chassis containing anywhere from a few dozen to several hundred physical disks.

With both SAN and NAS shared storage models involving a many-to-many connection scheme between the physical drives and logical disks, the full map of a complex SAN configuration can become quite the atlas!

Each of these links in the chain will affect the end-to-end performance of a shared storage system and the diverse possibilities that lie in each possible portion mean that shared storage configurations can vary to an astonishing degree.

In **Figure 2**, you can see a breakdown of some of the various components and configuration permutations of a shared storage system. This visualization of the potential complexity involved should effectively demonstrate the daunting nature of determining a storage system's end-to-end performance, as well as where its bottlenecks exist.

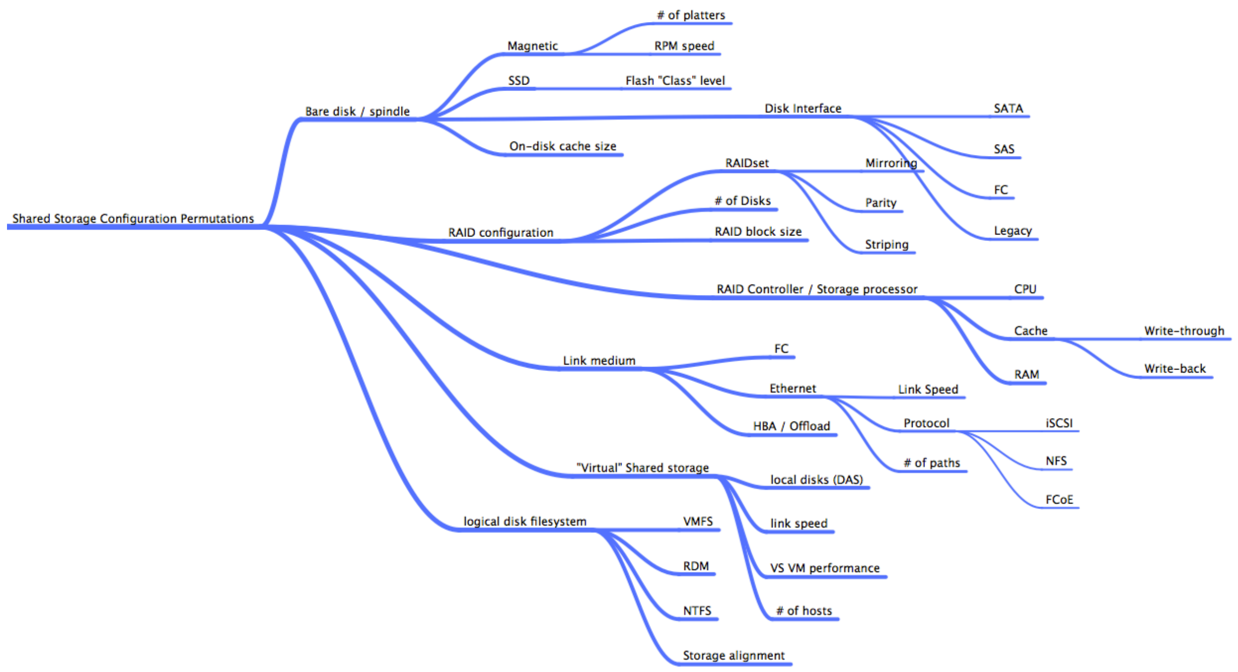


Figure 2: Possible Storage Configuration Permutations

Troubleshooting poor performance in a shared storage system can often feel like trying to search the inside of an oil tanker with a penlight. And while the specifics of what is causing slowness can be innumerably varied, the source or sources of storage limitations typically break down to a few major areas. These are where storage admins generally look:

- Hardware or system failures
- Communication "traffic jams" to the storage device
- Surges in storage I/O load demand
- Inefficient configurations for required throughput
- Application or OS-level inefficiencies

Failures in the chain of storage will cause at worst a complete outage, or at the very least a disruption in the performance of the complete system. For example, disk failures will place a LUN into a “degraded” state where the system attempts to work around the missing disk or disks. Degraded arrays will almost always be significantly slower in responding to IOPs and degraded LUNs might no longer have their redundancy and could exhibit data loss if further failures occur, so be sure to replace the failed disks ASAP. In larger arrays, you can see several disks fail each day! Plus, once a replacement disk is inserted, the array must be rebuilt. This causes the storage processor and all disks in said array to have an additional task to perform on top of whatever normal I/O load is being placed on it. This will slow down the responsiveness of the system even further.

There can also be failures in the communication between the hosts and the storage device. Loss of connectivity on redundant links will cause a reduction in the maximum amount of bandwidth via the SAN/NAS, or even sever connections entirely, causing outages or failover.

The storage devices themselves can also experience outages and failures, whether it is in the form of a storage controller failure, an issue on the backplane circuitry, or any other hosts of potential breakdowns. These failures are, admittedly, very rare due to redundant systems.

Generally, any good storage system has redundancy throughout, allowing for multiple components to fail without bringing production operations to a standstill. For this reason on top of the fact that failures are rare, this tends to be the least likely cause of performance problems in a shared storage configuration.

Outside of system failures, excessive I/O loads on a perfectly functional storage system can create higher latencies and therefore cause poor application performance. While this is the most obvious potential issue, it is also one of the most difficult to diagnose. Closely monitoring I/O load metrics such as IOPS and MB/s are critical for determining where heavy loads are coming from and how to most appropriately respond to those loads.

Visibility into the various “links in the chain” to determine the source of these metrics is also critical in narrowing down which portion of the storage system is taxed most heavily. Is the LUN overloaded, or are the iSCSI links saturated? Are the storage processor’s CPU’s unable to keep up, or are the host bus adapters inside the servers being pushed to their limits? If iSCSI or another Ethernet-based connection employed, are jumbo frames turned on at every step of the way? Very frequently, VM Admins don’t realize that even in vSphere 4.1, setting up multipath iSCSI with jumbo frames involves a lot of legwork in the ESX console before it works fully! This is being improved upon in 5.0 to be more GUI-centric, but double-check your work using the guide here to make sure:
http://www.vmware.com/pdf/vsphere4/r41/vsp_41_iscsi_san_cfg.pdf

Occasionally, the bottleneck might be within the application or OS, and not in the storage environment at all. For example, poor I/O performance on a database server might be due to tiny growth increments, which cause fragmentation. Another common (but decreasingly so) example is disk alignment. If the clusters of a partition are not aligned with the blocks of the disk (either physical or virtual) then reading one cluster may end up requiring access to up to three chunks of the LUN. Under extreme conditions, a misaligned disk can cause up to a 40% degradation in I/O performance! Most modern operating systems don't exhibit this problem, and typically it only occurs in legacy deployments.

In the end, working out bottlenecks in the chain involves figuring out which component is waiting for the other to finish its job and send the information along. Any time the average total latencies of I/O operations exceed 20ms, the applications and the users on them will begin to notice degradation of performance. As this latency increases, performance will only get worse, potentially reaching the point where the delay in reads and writes will cause the applications or OS's hosting them to time out and give up on their attempts to access the disk. In virtual and non-virtual systems, these situations are often logged as "aborted I/O commands," and they can cause serious high-level errors if proper handling for I/O is not implemented within the applications.

Measuring the I/O latency is most often done from within the storage. Tools from storage manufacturers or third-parties will grant visibility into the wait times for IOPS and can help pinpoint the source of the slowdown from within the storage system. However, these tools often overlook the guests themselves and their point-of-view. Measuring storage latency at the server (or VDI) is most important, since latency inside the array doesn't always account for the links and protocols which connect that the disks to all of the systems communicating with them. Looking deep into the storage device will help show which LUN or controller is having the most trouble, for example, but it will not help track down the fact that one of the network connections is saturated, causing the traffic in the SAN to bottleneck. So make sure that whatever methodology you are using to monitor disk performance incorporates a full view of latency end-to-end with the goal being to keep those numbers as small as possible.

What does all this mean? It means storage is complicated! Despite the simplicity of the concept to store a byte in a remote location and being able to access it from many places, the implementation of such becomes a mechanism of underappreciated complexity.

IOPS as a measurement of disk I/O

So, back to IOPS. Where does this measurement metric come into play, and how does it affect the overall picture of disk I/O? Can your storage system's performance be measured in IOPS effectively? Well, yes, but only in part. A better question to ask is, "What do you define as performance?" Are we talking about the maximum I/O potential of a storage system, or the responsiveness of the storage to the demand being placed upon it?

IOPS are an effect of a storage system's performance. Better performance on a more expensive Storage Array means more potential IOPS. Easy, right? Well here's where things become tricky:

An idle storage system has zero IOPS. As load increases on a storage system, the IOPS go up. As the load continues to increase, bottlenecks within the system will cause latency to rise and with more time between each I/O, the number of IOPS will eventually plateau. IOPS can therefore be an indicator of load under ideal circumstances, but one cannot simply say that because a Storage Array or Filer is showing a certain number of IOPS that it is performing well. It's not until the latency from those IOPS is examined that we know the storage has reached its saturation point and our applications have begun to suffer.

Storage performance affecting IOPS

As shown in **Figure 2**, there are many, many ways to build a storage system. And nearly all of these permutations will have an effect upon the number of simultaneous inputs and outputs that can be executed.

For example, let's say a single spindle can perform X number of IOPS. However, if that disk is now striped with a second disk inside an array, this increases the amount of possible I/O by giving the potential to read and write to both devices simultaneously. Thereby giving us X * 2 total IOPS available. Correct? Well, sort of. Even though the I/O is going to two disks at once, thereby increasing the total amount of data that can be written simultaneously, the application layer does not see this. The storage processor is transparently handling the transfer to both disks, and with the increase in bandwidth, can potentially handle more IOPS, but it isn't as simple as pure multiplication.

The process of encapsulating I/O also involves overhead on the part of the storage processor itself, so while adding more spindles to an array will bring about more available I/O capacity to a point, eventually the overhead becomes so great that all benefit of the parallelization is lost. This concept holds true throughout computing, so I won't address it in-depth in this article. For additional reading on the concept, point your browsers here: http://en.wikipedia.org/wiki/Parallel_computing

The potential bandwidth of a storage system also affects IOPS, regardless of an I/O's size or nature. The more bits that can be sent or received per second, the more IOPS. The end-to-end performance of the shared storage system will ultimately affect latency, and the higher the latency, the lower the IOPS.

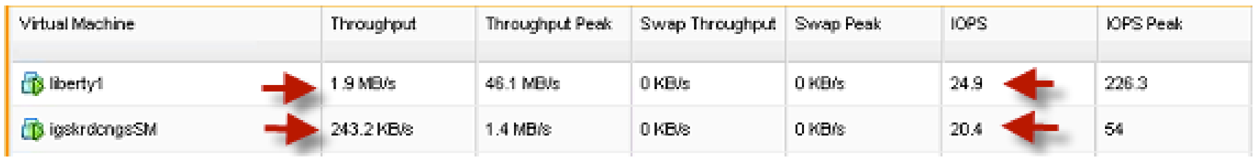
IOPS affecting storage performance

Now let's look at the situation from the other side. Reading or writing to a storage device means that those IOPS are placing load upon the network links, taking up CPU cycles on the VMs, servers, storage processors and HBAs, and making the heads of the spindles move to various locations around the disks. This means that if another system wants to utilize that same shared storage device, it will need to work within the boundaries of what remains. The disk heads will probably now have further to travel in order to read or write the next time around and the network links only have so much bandwidth remaining at that moment, etc. This means that the more IOPS that are being pushed through the storage, the slower the response time will be for other systems trying to access it. Even if one system has a relatively low demand on the disk, another with high demand will cause that low-demand one to have slower performance waiting with its foot tapping for the I/O request it sent to come back from the SAN.

Not all IOPS are created equal until they enter the storage system

The metric of I/O's per second is one that involves several caveats alluded to earlier in this paper. What size are the I/Os themselves? What percentage of them consists of read operations vs. write operations? Are they seeking data that's likely to be read from sequential areas of the storage or are they utterly random?

Figure 3 is a screen capture from just one environment where the disconnect between IOPS and total I/O load is markedly visible.



Virtual Machine	Throughput	Throughput Peak	Swap Throughput	Swap Peak	IOPS	IOPS Peak
liberty1	1.9 MB/s	46.1 MB/s	0 KB/s	0 KB/s	24.9	226.3
igskrdcngsSM	243.2 KB/s	1.4 MB/s	0 KB/s	0 KB/s	20.4	54

Figure 3 - Varying degrees of disk load but consistent IOPS

Notice the fact that the first VM is showing only 22% more IOPS than the second VM, yet it's showing 800% more disk throughput! Looking at throughput shows that the first VM is vastly heavier in disk I/O, yet if we were only considering IOPS, these two VMs on the same datastore would be almost indistinguishable between each other in their I/O needs. If these VMs have high disk latency (waiting long periods for their I/Os to return) how would you then determine which VM should be granted a more dedicated or higher-performance LUN for its operations in order to reduce that latency? Purely based on IOPS, it would practically be a coin toss. But since one VM is writing 64KB blocks and the other appears to be writing 12KB blocks, the difference is much more drastic.

Just looking at block sizes alone, a storage system's I/O capacity will vary greatly even with the same number of IOPS. **Figure 4** shows the kind of curve involved with I/O operation sizes and the number of them, which can be simultaneously handled. In short, the bigger the IOP, the fewer of them that can go through the entirety of a storage system.

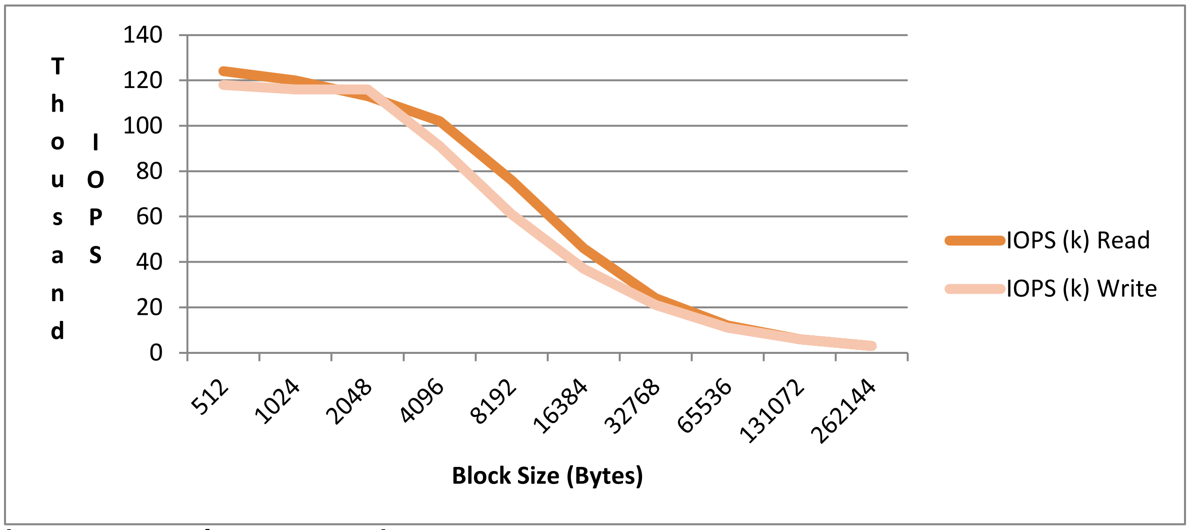


Figure 4 - Example IOPS capacity on a Texas Memory Systems SSD SAN

Conversely, the throughput potential of the entire system actually *increases* with larger block sizes. Much like enabling jumbo frames on an Ethernet connection can decrease overhead and improve maximum throughput for larger data transfers, the same can be true for disk I/O in certain circumstances. On the same TMS SAN, you can see this illustrated in **Figure 5**.

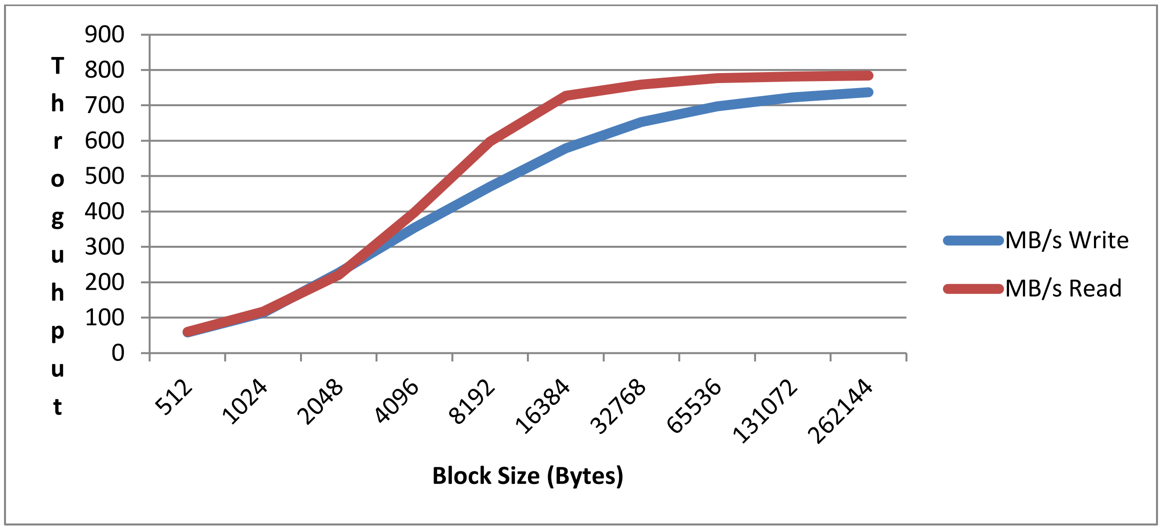


Figure 5 – Example Throughput Capacity on a Texas Memory Systems SSD SAN

Note that the total amount of I/O data throughput (essentially, the storage bandwidth) of a storage system plateaus at a certain block size and does not continue to improve. This is a perfect example of how the principle limitation of a storage system can be the total amount of bytes that can be written to or read from it each second and not so much the number of times a read or write operation can be executed on that system per second.

It's important to note that this differentiation ends once an I/O packet enters the physical storage device itself. Once inside the storage processor, the block sizes are consistent. However, when examining the end-to-end performance of a storage solution, this differentiation is vital.

So when trying to design and configure storage, do you focus on maximum potential I/O's or maximum throughput? The answer will depend on the following factors:

1) The class of operation
    a. Higher performance expectations and SLA's generally will demand low latency numbers, so storage must be not only optimal, but capable of sustained operation at high loads.
2) The type of data being accessed
    a. Large quantities of tiny files or granular database operations will benefit from systems that perform better with small block sizes, yielding greater IOPS for when throughput is secondary.
    b. Larger files or data access taking place in bigger chunks won't take advantage of smaller block sizes, and therefore will not benefit from an IOPS-oriented design, instead needing top performance for sustained throughput.

Let's take a VDI infrastructure for example:

In most virtual desktop deployments, disk I/O tends to be very heavy during initial boot-storms with close to 99% of the IOPS being reads from the boot image(s) that are very similar to one another. (Powering on a bunch of virtual desktops that all run the same operating system at the beginning of a work day.) Then, during normal operation the majority of IOPS are frequent, small, but very random writes alongside random reads. (Periodic saving of work files, writes to web browsing cache folders, email client activity when receiving messages, etc.) This kind of activity benefits heavily from caching whether on the spindles themselves, in the storage controller, or the HBA. During the boot periods all of the reads are from very similar or identical images, so caching means only one read IO from the spindles is needed for each sector, leaving the bottleneck to be the maximum throughput speed of the fabric to the storage array or Filer, provided the total caching size is large enough to store the full “golden” boot image within memory.

Once booted up, the virtual desktops issuing write commands will also benefit from write caching, allowing extra time for the spindles to keep up with receiving the I/Os. A 4GB write-back cache could provide space for over one million 4KB-sized I/O blocks (a common block size,) giving ample time for them to be written to disk in between bursts of disk activity. Once again, the bottleneck would usually come down to the fabric between the storage and the hosts. So in the case of VDI (often touted as a very IOPS-intensive function,) a Storage Array or Filer with ample caching would support a great deal of VDI-oriented IOPS before suffering any kind of slowdown either from the cache filling up.

Storage architects will quickly (and correctly) note that any sort of caching will only improve I/O performance at burst speeds, and primarily for write operations. Read operations do not always benefit from caching because they depend on the data already existing in the cache’s memory, either from being read previously or from an algorithmic selection of data that is likely to be read in the near future. (The percentage of success in utilizing a cache resource is known as its “hit rate”.) So while this might work ideally in the case of VDI, the same configuration may not do nearly as much good for a large-scale database deployment.

Making the Most of IOPS

With all of the complexities involved with storage, the divergent nature of throughput against IOPS, and the fact that no two IOPs are the same, does this mean that IOPS is a worthless metric? Far from it! It's merely one of many pieces of the puzzle of disk I/O. It also means that consistency is key, and that all the factors involved need to be accounted for in order to determine the performance requirements of an infrastructure, as well as the capabilities of a storage system to handle those requirements.

Consistency needs to come from the perspective of the storage vendor, so that when a system architect is looking to choose a storage solution, they have an even playing field. If a vendor promises a system capable of 1 million IOPS, and all of those IOPS are sequential, read-only with 100% cache success, and bursting for no longer than 10 seconds, then that information is at best, not helpful, and at worst, false advertising. An industry standard for overall I/O measurement is something that I believe the community should call for, but in the meantime, following these guidelines can help keep things in perspective when choosing a shared storage solution:

1) Assume the worst-case scenario and avoid over-simplification

Make sure that the storage vendor provides detailed information about the number, size, and type of IOPS the system can handle. Storage manufacturers tend to optimize their hardware and software for 512 byte transfers, to maximize their advertised IOPS rate. But if IOPS aren't as important as throughput for your application, this won't be optimal. Plus, 4KB or 8KB transfers are far more realistic to encounter in real-world applications.

Whenever possible, ensure that the numbers advertised assume a 0% cache hit rate, eliminating potential false performance improvements due to more rare "ideal" circumstances.

Arm yourself with information from application vendors ahead of time to work out what kind of I/O requirements you're going to be expected to support within the infrastructure. Lots of tiny I/Os vs. a small number of very large transfers will place very different requirements on your storage.

2) Hold each solution to the same standard

It's vitally important to determine the response time requirements of your applications in order to gauge the latency tolerances for your solution. The easiest way to do this is by testing in a POC environment, but even if you're unsure of the exact I/O needs of your application, use the same measuring stick between vendors and models.

Keep block sizes, stripe sizes, drive technologies and spindle counts consistent to narrow down the focus to the storage processor(s) and interfaces. Very frequently, Vendor A will use the exact same disks inside the Storage Array as Vendor B. Your focus, therefore, should be on working out how well their system can handle those disks.

3) Diversify, Diversify, Diversify

Don't put all your eggs in one basket. If you have a complex environment with variable storage performance needs, you should not be held to just one storage solution or model. Even if you keep a consistent vendor, focus on application-level delivery when determining the storage configuration that will work best for you.

And even if you want to keep a consistent vendor AND model, diversify the storage itself! Nearly all storage systems have provisions for more than one type of disk, storage processor, and communication medium.

Once a storage system is in place, following best practices in allocating space within it is critical to getting the most out of the deployment. For instance, when configuring LUNs it is important to divide and conquer. Before you put all of your disks into one logical array and expect great performance, make sure that your storage processor is equipped to handle that kind of configuration! Depending on the number of disks, the parity calculations alone could push your storage processors to 100% even with the lightest of I/O loads.

It is better to split things up. Put your I/O-light apps on inexpensive disks in parity configurations, and dedicate high-performance drives in their own LUNs for databases or other I/O–intensive operations. This can almost always be accomplished inside the same physical Storage Array.

Storage vendors will have their own technologies and methods for assigning the right amount of disks in the right configuration for best performance and most efficient distribution of available space. Make sure to work closely with the manufacturer whenever possible, since even someone with a great deal of storage experience can be taken by surprise.

4) Check your (and their) work.

Verify that the promised amount of I/O capacity lines up with what should be theoretically possible given the hardware configuration proposed. Calculators like the one here by Marek Wołynko can help:
http://www.wmarow.com/storage/strcalc.html

Utilize tools such as I/O Meter: http://www.iometer.org/ to push storage systems to their limits and see how well they perform under high-stress loads before they end up being put into production.

Once in production, closely monitor I/O responsiveness from the perspective of the systems utilizing the storage. Keep an eye on your latency values. No IO operation should be taking longer than 50ms to round-trip within the storage environment in order to maintain the best performance and even tighter tolerances will be needed for higher-tier operations.

Conclusion

It is clear that IOPS as a term for storage performance is here to stay, regardless of any limitations that may exist with its use. The best way to keep on top of what it means to you and to your infrastructure is to remain informed. This whitepaper began as a blog post just about IOPS, but this topic is so enormous that I am going to keep this as a “work in progress” that I will keep updating with time.

Good luck!