Table of Contents

Overview .................................................................................................................. 4

Typical Storage Considerations in a Virtualized Desktop Environment ............. 5
  Desktop Operating System Considerations ......................................................... 5
    What Windows Wants ......................................................................................... 5
    Helper Technologies ......................................................................................... 5
    SuperFetch ........................................................................................................ 5
    Indexing ............................................................................................................ 6
    Disk Defragmenter ............................................................................................ 6
  Windows Image Optimization ............................................................................. 6

Performance and Capacity Considerations ......................................................... 7
  The Importance of IOPS ...................................................................................... 7
  Workload IOPS Assumptions ............................................................................. 8
  Workload Read/Write Ratio ............................................................................... 9
  RAID Penalties .................................................................................................. 9
  Calculating IOPS Requirements ....................................................................... 10
  Random vs. Sequential Workloads and the I/O Blender Effect ....................... 11

Shared Storage Protocol Considerations ............................................................. 12
  Maximum Transmission Rates by Protocol ....................................................... 12
  Multipathing and Balancing Throughput to Shared Storage ............................ 12
  Block vs. File-Level Storage Protocols .............................................................. 13
    Block Based (FC, iSCSI) .................................................................................. 13
    File Based (NFS) ............................................................................................ 13
  Storage Fabric Considerations ......................................................................... 13

Capacity and Sizing Considerations .................................................................. 14
  Floating or Dedicated Desktops ......................................................................... 14
  Full, Linked, and Array-Based Clones ............................................................... 14
    Full Virtual Machine Clones .......................................................................... 14
    Linked Virtual Machine Clones ....................................................................... 15
    Array-Based Virtual Machine Clones ............................................................. 16
  VCAI Virtual Machine Linked Clones (Tech Preview Feature of Horizon View 5.2) ................................................................................................................. 17
  Sizing Guidelines for Full and Linked-Clone Desktop Pools ......................... 17
    Virtual Machine Swap ..................................................................................... 17
    Full Clone per Virtual Machine Calculation ................................................ 18
    Linked Clone per Virtual Machine Calculation ............................................. 18
  Persona, User, and Shared Data Considerations ............................................. 18
Overview

This document will provide the reader with a high-level overview of the challenges associated with desktop workloads in a virtualized environment, and offer design considerations in managing these issues. The focus is on the typical challenges associated with the storage subsystem of a virtualized infrastructure, with respect to performance capacity, and operational considerations.

Where possible, this document offers more than a single solution alternative to an issue, in order to provide the reader flexibility when considering common design choices. The document will also maintain an element of agnostic analysis.

By completion of this document, the reader should have a clear overview of the challenges and requirements that an IT architect faces when designing and implementing a storage strategy for a virtual desktop infrastructure. The reader will also have an understanding of the features in VMware® Horizon™ View 5.2 that assist in the storage sizing and performance aspects of a virtual desktop infrastructure.

In the Appendix, the authors take the reader through a user sizing example specific to Horizon View 5.2, following VMware recommended practices.
Typical Storage Considerations in a Virtualized Desktop Environment

This section establishes and explains several of the typical issues IT architects face when setting out to design a virtual desktop infrastructure (VDI) solution.

The reader might ask, “Why do I need to consider the storage requirements? I should be able to size a deployment based on capacity requirements, pick my favorite vendor and the correct number of disks, and add to cart.” At some point in the not-too-distant future this may be the case, as technology vendors have created many truly innovative solutions—some of which are investigated in greater detail in this document.

In order to have success in design, operation, and scale, however, IT must be diligent in the discovery and design phases, to make sure they have a strong methodology and a plan to adapt or refine certain elements when technology changes.

Desktop Operating System Considerations

To fully understand the implications of virtualizing and consolidating multiple desktop operating systems on a single host, first consider how a typical desktop OS behaves with respect to storage and its associated subsystems.

As most architects will be deploying a Windows-based operating system (OS or guest OS) such as 32- or 64-bit Windows 7, this paper takes a high-level tour of the important aspects in your design that are specific to the Windows operating system. First consider the OS itself, its I/O requirements, and then the sizing requirements for a virtual desktop infrastructure.

What Windows Wants

Bear in mind that any desktop operating system, including Windows, is designed without consideration for virtualization technologies, in particular when it comes to the storage subsystem. Windows has been designed to interact with a locally connected magnetic disk resource.

The OS expects at least one local hard disk dedicated to each single instance; and has complete control from the device driver upwards with respect to the reading, writing, caching, arrangement and optimization of the file system components on the disk. When you install the OS into a virtual machine running on a hypervisor, and particularly when you run several virtual machines simultaneously on that hypervisor, you should be cognizant of certain factors that will affect the operation of the operating system.

The first set of considerations is the helper technologies built into the OS itself.

Helper Technologies

It is worth noting that Windows also has some features, enabled by default, which can optimize and accelerate the reading and writing of files from the local disk. While they are of obvious benefit in a 1:1 relationship between OS and disk, when you virtualize and consolidate multiple guest OSes on a single hypervisor you multiply the IOPS overhead, which the underlying storage subsystem must contend with.

Below are specific examples that illustrate these features.

SuperFetch

As part of the logical Prefetcher service in Windows 7, the SuperFetch feature requests boot and application startup traces from the logical Prefetcher, post-processes the trace, and writes the information to a file in the %SystemRoot%\Prefetch directory. When Windows boots or an application is launched, the Prefetcher looks in this directory to see if a trace exists. If it does, it calls NTFS to prefetch any metadata file references, reads the contents of the files, and finally opens the files that are referenced.

The prefetch service also organizes the list of boot or application startup files in the order that they are referenced in the layout.ini file, also found in the %SystemRoot%\prefetch directory. However, during system idle times the Prefetcher launches the system disk defragmenter with command-line parameters
to defragment, based on the contents of the `layout.ini` file, rather than launching a full defragmentation.
This process attempts to find a contiguous area on the disk to place these files in order to speed up subsequent
boot or application load times.

**Indexing**
The Search component of Windows is designed to accelerate searching for content in either the local disks or
network attached drives. The most important component, the indexer, uses `SearchIndexer.exe`,
which hosts the indexes and the list of URIs (uniform resource indicators) that require indexing.
`SearchProtocolHost.exe` hosts the protocol handlers. `SearchFilterHost.exe` hosts the IFilters
and property handlers to filter metadata and text content. The indexer crawls the file systems at startup, then
monitors for changes that may affect its index.

**Disk Defragmenter**
Microsoft has updated the disk defragmenter service in Windows 7 to include more files for scanning and
reallocation, including metadata for the NTFS file system. The purpose of this is to periodically scan the disks
attached to the guest OS, and relocate files into contiguous areas of the disk to reduce access times. The
defragmentation process runs periodically as a low-priority background task in the OS. Generally you should
attempt to eliminate unnecessary I/O from the guest OS, especially when you can deal with file placement and
other disk management tasks at the hypervisor and storage-array level.

**Windows Image Optimization**
For these and other performance-related reasons, VMware recommends optimizing the base system image in
accordance with VMware best practices. See the References section for additional information on performing
these optimizations.
Performance and Capacity Considerations

Storage considerations in a virtual desktop infrastructure have two dimensions: performance and capacity.

The Importance of IOPS

When you consolidate multiple virtual machines on a hypervisor, you should understand the typical storage performance a single OS expects, and understand that you will add contention to the storage subsystem with every subsequent guest OS that you host on that hypervisor. Performance with respect to storage is generally quantified in terms of IOPS.

The traditional formula for calculating IOPS is shown below:

\[
\frac{\text{Rotational Latency} + \text{Seek Latency}}{1000} = \text{IOPS}
\]

This document will revisit this equation later, and replace it with VMware-specific derivatives. However, to make sense of the calculation, consider a typical desktop PC configuration and look at the kind of hard disk it ships with. A typical multipurpose desktop PC today will ship with a 7200RPM SATA3 hard disk, which should yield at least 75 IOPS, to be used at the operating system’s discretion.

Table 1 shows common disk rotational speeds and their respective IOPS estimate.

<table>
<thead>
<tr>
<th>ROTATIONAL SPEED (RPM)</th>
<th>IOPS ESTIMATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD*</td>
<td>6,000</td>
</tr>
<tr>
<td>15k</td>
<td>175</td>
</tr>
<tr>
<td>10k</td>
<td>125</td>
</tr>
<tr>
<td>7,200</td>
<td>75</td>
</tr>
<tr>
<td>5,400</td>
<td>50</td>
</tr>
</tbody>
</table>

*SSD included for comparison

Bear in mind that you are not taking into account any kind of shared storage infrastructure yet. You must have a reference point for when you start to introduce the concepts of shared storage, interconnects, RAID penalties, and cache considerations.

The next logical step in your storage design process is to understand what a typical or normal Windows workload requires in terms of IOPS. You should also become familiar with the concept of steady-state IOPS requirements. This includes how much I/O you must have when everything is up and running in business-as-usual mode; and also storm IOPS requirements, which you typically encounter in large-scale provisioning and deployment activities and virtual machine boot scenarios. Although storm conditions are much less frequent, the IOPS requirements for these situations can be an order of magnitude larger than in steady-state mode. Storm situations can overwhelm an inadequately sized storage system, rendering the environment unusable.
Workload IOPS Assumptions

It is possible that every department in an organization has different IOPS demands they make of their virtual desktops. IT architects, however, must have general estimates on which to base their initial IOPS sizing. Their estimates can be refined later in the design process with real-world data gathered from proof-of-concept and pilot activities.

As a rule of thumb, the IT industry breaks down IOPS profiles into four discrete types of users, as shown in Table 2.

<table>
<thead>
<tr>
<th>USER CLASSIFICATION</th>
<th>SIMULTANEOUS APPLICATIONS IN USE</th>
<th>VIRTUAL MACHINE CONFIGURATION</th>
<th>IOPS REQUIREMENTS PER USER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task User (Light)</td>
<td>Limited 1–5 apps light use</td>
<td>1vCPU 1GB RAM</td>
<td>3–7</td>
</tr>
<tr>
<td>Knowledge Worker (Medium)</td>
<td>Standard productivity 1–5 apps regular use</td>
<td>1vCPU 1GB RAM</td>
<td>8–16</td>
</tr>
<tr>
<td>Power User (Standard)</td>
<td>Compute intensive &gt;5 apps regular use</td>
<td>1vCPU 2GB RAM</td>
<td>17–25</td>
</tr>
<tr>
<td>Power User (Heavy)</td>
<td>Compute intensive &gt;5 apps intense use</td>
<td>2vCPU ≥3GB RAM</td>
<td>25+</td>
</tr>
</tbody>
</table>

Where a typical call center worker with a standard multipurpose physical desktop has 75 IOPS at their disposal, replacing this with a virtual machine sized for a maximum of 7 IOPS might have disastrous results, in the form of unhappy users and a failed virtual desktop project.

In the absence of absolute numbers from your existing user estate, the table above is the best guidance available. In practice, unless you are designing a virtual desktop infrastructure to only support the lightest of users, you should be sizing for Power User (Standard) and Power User (Heavy).

The majority of virtual desktop deployments are intended to provide a primary computing environment for the end user—in other words, a desktop replacement. You can reduce the raw IOPS requirements for Windows with optimization and tuning techniques, which this paper covers later. You must design for performance in all but the most specific use cases, such as a call center agent who uses a single Web-based application.

If you intend to replace your users’ physical PC device with a virtual desktop, it is critical that the performance of the virtual machine is equal to or better than the device you are replacing. If you intend to replace the CEO’s ultrabook, they might expect 6,000 IOPS of performance from their virtual desktop. This is an exception; a pragmatic design might incorporate tiered service for users expecting this kind of special treatment, for example, a separate high-performance pool for the demanding user.

In real-world customer deployments, a typical primary use virtual desktop will have 2vCPUs with 2GB RAM or more. There is often more when a 64-bit OS is used—64-bit is almost always used primarily to increase the amount of user mode memory available in the virtual desktop. You’ll also see in the Capacity and Sizing Considerations section of this paper why vCPU and vRAM are important to the capacity per virtual machine requirements.

As an average, users run at least five applications simultaneously. In addition, it is likely that they have requirements for rich media, such as video or unified communications, running in the virtual desktop—if not now, then certainly in future iterations. This use case has become the rule rather than the exception, and in lieu of assessment exercises it is a great starting point for your design.
Now that we have established a baseline, below is an introduction to other concepts you should consider when establishing IOPS requirements.

**Workload Read/Write Ratio**

When considering Windows as your primary workload for storage design, it is important to fully understand the I/O characteristics of the OS and your typical users. In particular you should look at the ratio of read I/O operations to write I/O operations in both storm and steady-state conditions.

The Windows file system NTFS frequently issues small random writes to maintain its metadata transaction log, a mechanism used for consistency and recovery from an unexpected system halt.

In addition, you have application I/O and user profile I/O, which increase the workload. Everything must interact with the file system for general operation, which increases the overall I/O load on the system.

Taking this information into account, along with experience and your own assessment numbers, typically you can surmise that virtual desktop workloads demonstrate a higher number of write I/O operations than read I/O operations when operating under steady-state conditions. However, there are certain storm conditions that result in read I/O that can be several orders of magnitude greater than the write I/O. The typical example is a virtual machine boot storm.

Read I/O caching and acceleration techniques have been available for years, and now there are features built into most virtualization platforms to account specifically for this particular scenario. This paper covers these approaches later in detail, as you must consider both the steady-state and storm requirements for your final design. However, for now we are only concerned with steady-state workload requirements.

A starting point for you to use is the read/write I/O ratio in Table 3.

<table>
<thead>
<tr>
<th>STEADY STATE</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Read I/O %</td>
<td>Write I/O %</td>
</tr>
<tr>
<td>40</td>
<td>60</td>
</tr>
</tbody>
</table>

Table 3: Read/Write I/O Ratio

Some VMware customer deployments experience read/write I/O profiles with far higher weight on the write I/O side, in some cases as high as 80–90 percent. Use an assessment tool to reinforce your design, and if necessary, modify the inputs to suit the user profiles in the enterprise.

**RAID Penalties**

Up to this point this paper has considered the performance characteristics only of the guest OS on a single, dedicated disk. With a virtualized environment, VMware introduces the concept of shared storage systems and disk targets comprised of RAID volumes, the latter applicable to both shared and local disk subsystems.

While a deep discussion of RAID is beyond the scope of this document, at this point you should consider the concept of RAID penalties in order to accurately size for your workload IOPS requirements. Simply, RAID introduces a write penalty that differs with the type of RAID set you are using for the target volume. This is also compounded with parity checking and other considerations beyond the scope of this document. For further reading, see the References section for blogs that discuss these issues in greater technical detail.

RAID penalties are especially critical to consider, as the workload you are sizing for will potentially have a greater requirement for write I/O than for read I/O. Table 4 shows the write penalty introduced when using common RAID configurations.
In Table 4, you can see that in an example RAID-6 configuration, for every write operation there are six I/Os needed to complete the operation. This is the write penalty that you must accommodate in your calculations.

### Calculating IOPS Requirements

As mentioned earlier with regard to the standard formula for calculating IOPS, this paper changes the formula slightly to the following variant, as you now know the target IOPS you will be designing for:

\[
\text{Target IOPS} \times \text{Read I/O\%} + (\text{Target IOPS} \times \text{Write I/O\%} \times \text{RAID Penalty}) = \text{IOPS}
\]

Using a real-world example, if you size a single virtual machine workload at 25 IOPS with a read/write ratio of 40/60, using a RAID-5 disk volume your equation looks like this:

\[
(25 \times 0.4 + ((25 \times 0.6) \times 4)
\]

Which is:

\[
10 + 60 = 70 \text{ IOPS}
\]

Your requirement for 25 IOPS on the front end is actually a back-end requirement of 70 IOPS, when taking RAID-5 as an example.

The reason why this is so critical is that if you sized your storage performance requirements for 500 VDI virtual machines based only on front-end IOPS numbers, you would arrive at a target steady state of 12,500 IOPS. However, if you chose to deploy this on a RAID-5 volume, your actual back-end requirements would work out as:

\[
((12,500 \times 0.4) + ((12,500 \times 0.6) \times 4)
\]

or:

\[
5,000 + 30,000 = 35,000 \text{ IOPS}
\]

Here you end up 22,500 IOPS short of delivering on the target for all 500 of your users. This extreme example illustrates the importance of sizing correctly, considering your virtualization and storage subsystems.

While caching technologies available from storage array vendors can serve to absorb many of these write penalties, their effectiveness varies by vendor. This topic is beyond the scope of this document; but this document includes excellent resources for further research in the References section.
Random vs. Sequential Workloads and the I/O Blender Effect

As mentioned earlier, over the years operating systems, in this case Windows, have integrated several techniques in an attempt to serialize I/O with respect to disk read and write operations. The benefits of this are logical: if reads and writes occur to contiguous areas of the disk, seek times, and therefore latency, during these operations are kept as low as possible.

Again, in a 1:1 relationship between the OS and disk, this is desirable behavior. However, when you consolidate multiple virtual workloads on your hypervisor, multiple streams of serial I/O become fragmented and are essentially converted to a random pattern. The simplified explanation for this anomaly is that the hypervisor resource scheduler multiplexes the I/O streams, in order to load balance requests to the compute, networking, and storage subsystems. The effect of this is amplified when you consider virtual desktop workloads. As discussed, virtual desktop workloads are more likely to produce frequent random write I/O to their storage, and are generally more write intensive.

It is important to be aware of the blender or randomization of I/O created by consolidating workloads on a hypervisor. Fortunately there are several tactical approaches you can leverage with technology to subdue these effects. Queuing and coalescing read/write I/O in front of the disk subsystems can be achieved with cache processing. This can help normalize the access patterns and mitigate the randomization effects. This paper discusses these techniques in greater detail in the Storage Platform Considerations section.
Shared Storage Protocol Considerations

VMware vSphere® supports several storage protocols. Your storage design can incorporate one, many, or all of the following choices:

- Fiber Channel Protocol (FCP)
- Fiber Channel Protocol over Ethernet (FCoE)
- iSCSI
- NFS

Several factors can influence your choice: for example, whether you are designing a greenfield site or repurposing existing infrastructure. Consider the requirements this paper has already discussed when you design for a virtual desktop infrastructure. In some cases, a hybrid approach might be necessary to satisfy the performance and capacity requirements of your design.

Maximum Transmission Rates by Protocol

Table 5 shows the maximum theoretical transmission rates by protocol, assuming a 10Gb Ethernet network infrastructure for the network-based protocols FCP, FCoE, iSCSI, and NFS.

<table>
<thead>
<tr>
<th>PROTOCOL</th>
<th>MAXIMUM TRANSMISSION RATE</th>
<th>MAXIMUM THEORETICAL IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fiber Channel Protocol (FCP)</td>
<td>8Gbps</td>
<td>16,384</td>
</tr>
<tr>
<td>Fiber Channel Protocol over Ethernet (FCoE)</td>
<td>10Gbps</td>
<td>20,480</td>
</tr>
<tr>
<td>iSCSI</td>
<td>10Gbps</td>
<td>20,480</td>
</tr>
<tr>
<td>NFS</td>
<td>10Gbps</td>
<td>20,480</td>
</tr>
</tbody>
</table>

Table 5: Maximum Theoretical Transmission Rates

Multipathing and Balancing Throughput to Shared Storage

When you look at the protocol performance above, it’s important to note that the theoretical IOPS numbers apply to a single communications path. If you take the example of 500 desktops of 35,000 IOPS, you can see that if you present these on a single path to the hypervisor, you will lack slightly more than 40 percent of the required IOPS to support your steady-state utilization target. In the real world this is a highly unusual configuration.

For your virtual desktop and server infrastructure, it is important to consider creating multiple paths to your target storage processors, and balance shared storage LUNs (logical unit numbers) appropriately in order to provide redundancy and performance.
Block vs. File-Level Storage Protocols

Below is an introduction to the differences between block- and file-based protocols with regard to your storage design.

Block Based (FC, iSCSI)
A block-based file system presents raw storage to your hypervisor as if it were a physical hard disk, albeit connected via a fabric, either FC for fiber channel or iSCSI for Ethernet. Typically, in VMware-based deployments you will calculate the appropriate LUN size. (See Capacity Sizing and LUN Allocation in the storage sizing example provided in Appendix A for more information.) Then you’ll create the necessary amount of LUNs; configure zoning for presentation; and finally, present block-level storage devices to the hypervisor, which are then formatted with the VMFS file system.

Block-level file systems are generally very flexible and allow various configuration options in a storage design, which ultimately gives you more scope for changes later. However, block-based file systems are generally more complicated to implement.

File Based (NFS)
In the same respect that block-level storage is flexible, file-based storage by comparison is often more straightforward to design and implement. A file-based system such as NFS can be easier to deploy with respect to a virtual desktop environment. The storage processors in the array themselves, rather than the hypervisor, handle the filesystem and access control operations. With the exception that you might have to set up permissions in advance, when these are in place you can present the NFS datastores to the hypervisor, and be ready to deploy virtual machines.

Storage Fabric Considerations
Although FC, FCoE, and iSCSI are block-based, only a fiber channel accesses the storage processors over a dedicated fabric. FCoE, iSCSI, and NFS all leverage an Ethernet network in order to access the storage processors.

Appropriate weight must be given to the Ethernet network architecture when designing specifically for storage access in any virtual desktop infrastructure design. Appropriate use of segmentation with VLANs, port density, subscription rates, and fault tolerance should be considered, along with allowing multiple paths and ample bandwidth from the hypervisors to the storage processors in order to satisfy the calculated steady-state I/O requirements.

There is no right or wrong approach at the filesystem level when designing for shared storage, as long as either option is fully explored and designed to meet the performance, scalability, and capacity requirements of your design.
Capacity and Sizing Considerations

After you have designed your methodology for estimating virtual desktop performance requirements, you must complete the relatively more straightforward task of capacity planning.

Floating or Dedicated Desktops

Immediately the IT architect is faced with a big decision, “Am I going to deploy persistent (dedicated) or floating (stateless) desktops to my users?” While there are arguments for both approaches, much of the decision process revolves around the operation and support of virtual desktops in a dedicated or floating model.

The merits of either approach are beyond the scope of this document. However, consider the advantages of operating a linked-clone desktop pool as opposed to a full clone, with specific respect to patching, updating, and lifecycle management.

Full, Linked, and Array-Based Clones

Next, you have to decide which kind of virtual desktop virtual machine you will deploy in your design. With specific regard to a Horizon View deployment, you have the option of choosing full, linked, or array-based clones of your master virtual machine image. It is important to consider the correct desktop type for your deployment, and there are several factors to weigh before deciding on a strategy.

Full Virtual Machine Clones

The first approach is to use full clones for your virtual desktop machines, specifically full-clone operations initiated by vSphere. You’ll see why this differentiation is necessary in the Array-Based Virtual Machine Clones section.

IT architects often leverage full clones for dedicated desktop pools, where the users expect to be connected to the same desktop virtual machine every time, and where specific software requirements dictate this approach. A good example is a developer desktop, as shown in Figure 1.

![Figure 1: Developer Desktop Full-Clone Operation](image-url)
In a desktop pool of full clones, the clones are identical in size to the parent image, which makes the sizing calculation straightforward. Your requirements for storage capacity may be an order of magnitude greater than if you were to use an alternative approach.

**Linked Virtual Machine Clones**

Virtual desktop infrastructure deployments with Horizon View often leverage VMware View Composer™ linked clones for their desktop pools. View Composer uses VMware linked-clone technology to optimize desktop storage space and improve image control. Linked clones act as unique pointers for each user to a single virtual machine master.

These linked-clone virtual machines each have unique identities, and can be powered on, suspended, or reconfigured independently of the master image. Linked clones can be refreshed at the administrator’s discretion, without affecting user data or settings, to ensure tight OS management and optimize storage resources, as shown in Figure 2.

![Figure 2: View Composer Linked-Clone Operation](image)

There are several advantages to using linked clones in your deployment. First, as this is a storage discussion, are the storage advantages. In simple terms, the linked clone virtual machines share a common replica image, and as such the linked clone itself can have as much as 70 percent less capacity requirement than the parent image. This can greatly reduce the overall storage requirements for a given deployment. For real-world sizing, assume a 50 percent reduction in the linked clone size relative to the parent or replica.

VMware vSphere 5.1 and Horizon View 5.2 can support up to 32 hosts on a VMFS- or NFS-based datastore, which means that you can create much larger linked-clone desktop pools than in previous versions. However, linked clones can grow in size over time, and you must give appropriate consideration to the expected growth relative to the refresh/rebase schedule in your deployment. Consider enabling the space-efficient sparse disk (SE Sparse disk) feature to reclaim unused space in the linked-clone virtual machine. (SE Sparse disk format is discussed later in this paper.)
Along with capacity reductions, you can realize several operational benefits when you use linked clones. Among these are much more efficient provisioning operations in the form of deployment of, refreshing, and recomposing desktop pools. Linked clones greatly simplify patch management and base image updates, and offer enhanced mechanisms for stateless or floating desktop pools, such as the ability to refresh or delete the linked clone-image when a user logs out.

Leveraging tiered storage in conjunction with Horizon View Composer linked clones gives you the ability to redirect user data to an alternative datastore. This allows the linked-clone virtual machine OS to be refreshed or rebased while preserving local user data, as you can snap out and snap in the persistent disk to the virtual machine.

You can perform similar optimizations with OS memory paging and temporary files, leveraging the disposable disk feature of Horizon View 5.2. Utilizing the disposable disk allows you to redirect transient paging and temporary file operations to a VMDK hosted on an alternate datastore. When the virtual machine is powered off, these disposable disks are deleted.

**Array-Based Virtual Machine Clones**

Some array vendors also offer an alternative cloning method outside of vSphere: block-based array cloning. In this method, the parent virtual machine is efficiently cloned by the storage processors in the array, in most cases using a native, block-level operation. This is shown in Figure 3.

Array-based clones are typically full clones of the parent virtual machine. However, in some cases the cloning operations can be an order of magnitude faster than using vSphere-based cloning. When the clones are created, they are powered up and customized using a process analogous to the physical process. In the case of Windows, an unattended Sysprep answer file customizes particular information, and generates a new SID for your desktop virtual machines. These newly created virtual machines are fully independent, with no dependency on the master image, and operate in similar fashion to a standard full clone.

When these operations finish, you should create a manual desktop pool and import the new clones. In some cases vendors have also automated this process. There are several use cases where providing array-based clones might be advantageous, particularly in environments where large pools of virtual machines are provisioned, then removed. A good example of this is a lab or classroom environment that experiences highly transient and disparate desktop virtual machine configurations.
VCAI Virtual Machine Linked Clones (Tech Preview Feature of Horizon View 5.2)
The View Composer Array Integration feature of Horizon View 5.2 leverages capabilities of VMware vSphere, as well as some of the NAS storage arrays that have the VAAI (vSphere API for Array Integration) NAS native snapshot capability.

This enables the creation of linked clones to be offloaded to a storage array, with View Composer still in control of provisioning operations. It is used in conjunction with linked-clone desktop pools and NFS datastores that are exported by the NFS array vendors, as illustrated in Figure 4.

VCAI clones leverage the best of both worlds from an operational and efficiency standpoint. View Composer manages the cloning processes initiated from Horizon View, while the VCAI-capable array offloads cloning tasks to the storage processors.

VCAI is still a Tech Preview feature. VMware does not support it in production deployments, and as such you should use it only as a reference option in your storage design. Furthermore, VMware only supports this feature on NFS datastores in conjunction with Horizon View, which might not be suitable for your storage design requirements.

Sizing Guidelines for Full and Linked-Clone Desktop Pools
When sizing per virtual machine, consider only full clones and linked clones with regard to your calculations. See Appendix A for a performance and capacity-sizing example.

Virtual Machine Swap
Another dimension of per-virtual-machine calculations is virtual machine swap files. There are two types of swap files to consider. First is the vRAM or virtual machine swap file (.vswp) stored with the virtual machine. It is equal to the amount of non-reserved vRAM allocated, or 100 percent of the allocated vRAM if not using memory reservations. Next, you have the secondary swap file or overhead swap file, which is created to accommodate operations when the host is under memory pressure.

For the virtual machine swap, consider reserving a proportion of the allocated vRAM in the virtual machine to balance the capacity overhead the swap file produces. For example, for a virtual machine with 2GB vRAM, consider a 1GB reservation to reduce the swap file size by 50 percent. Calculating the overhead swap file is covered in Appendix A.
Full Clone per Virtual Machine Calculation
The storage capacity required for a full clone virtual machine is simple to calculate using the formula in Figure 5.

![Figure 5: Formula to Calculate Full Clone Storage Capacity](image)

This will give you the storage capacity required per full clone virtual machine in your desktop pool.

Linked Clone per Virtual Machine Calculation
To calculate storage capacity for a linked-clone virtual machine, use the formula in Figure 6.

![Figure 6: Formula to Calculate Linked Clone Storage Capacity](image)

The replica size, equal to the master image size, is taken into account on a per-LUN basis. A replica of the master image is automatically placed into each LUN that you select to host the linked-clone virtual machines in your Horizon View desktop pool.

Capacity per linked clone virtual machine begins with the linked clone itself (50 percent of the replica size is a good estimate, as discussed previously). Then, add the amount that you anticipate the linked clone to grow to between refresh/rebase operations. (Twenty percent of the linked clone size is a good estimate—consider a greater percentage if refresh/rebase operations will be less frequent.) Finally, add the virtual machine swap and overhead.

When you use the persistent and disposable disk features, they must also be calculated for the overall capacity requirement.

Persona, User, and Shared Data Considerations
You must consider the final dimensions of the entire capacity requirement, which are persona or profile, user, and finally corporate data.

Unless you are deploying a completely new infrastructure, it’s likely that shared files and data are beyond the scope of your virtual desktop storage design and will already exist. However, if you are looking to deploy an architecture that leverages either standard Windows roaming profiles or a proprietary solution (for example, Persona Management with Horizon View) then you must allow for the added capacity requirements this will require. This applies in both full- and linked-clone architecture, as shown in Figure 7.
In many deployments, IT architects will incorporate different tiers of storage for specific data types. An example is to use high-performance VMFS datastores for the virtual machine OS and swap, and NFS for shared and user data. This does not imply that NFS performs less well than block-based storage. Rather, the management and operation of this type of data might be more suited to an NFS file system.
Consider the performance, capacity, and operational requirements in your final design. The example in Figure 8 shows a linked-clone virtual machine data layout with Persona Management, user data, and corporate shared data tiered on different datastores.

Figure 8: Linked-Clone Virtual Machine Data Layout
Storage Platform Considerations

Now that you have acquired the correct sizing guidelines for performance and capacity, the next step is to look at options for storage when it comes to deployment.

The storage landscape, with specific regard to virtual desktop workloads, is probably one of the fastest changing areas of enterprise IT today. The traditional choice of a standard, spinning, hard disk-based storage array is no longer the only option. In fact, most configurations today include caching, solid-state, or some kind of hybrid technology.

Combined with the leap in shared storage arrays technology, it is important when planning your architecture to consider the more recent appearance of converged appliances, virtual storage appliances, and flash architectures in direct attached storage.

As you know by reading the previous sections, virtual desktop workloads require the storage architect to balance performance and capacity requirements very carefully, in order to realize the best performance possible within a budget.

Figure 9 illustrates the deployment options available today.

**Figure 9: Available Storage Deployment Options**

**Direct-Attached Storage Considerations**

Direct-attached storage (DAS) architectures are increasingly more popular in virtual desktop design. This is primarily due to the adoption of the VMware reference architecture for stateless virtual desktops, which leverages local SSD disks for the virtual machine linked clones and swap. In a DAS architecture, there are tradeoffs with respect to the high-availability features of the hypervisor. Specifically with VMware vSphere, features such as DRS, Storage DRS, and HA require a shared storage architecture in order to operate. With a virtual desktop design, these features might not be as critical as for server workloads, as the infrastructure operates differently.

With a virtual infrastructure handling server workloads, IT architects can expect capacity demands to be more dynamic. You may require more resources during specific times of the day, week, or month, and can shift resources appropriately to accommodate bursty behavior.

In a virtual desktop infrastructure, architects can expect much more predictable demand for resources, so IT tends to design for 100 percent concurrency in terms of compute and storage in steady state. You can easily achieve a balanced design, with appropriate consideration for fault tolerance and failover capacity, using local storage. Local storage also keeps the I/O boundary within the server host itself, making sizing and scale-out operations far simpler by adopting a modular building-block methodology.
Enhanced VMware vSphere vMotion® capabilities in vSphere 5.1 also allow you to move a file by vMotion without the need for shared storage architectures. Look for more information in the References section.

**Flash-Based Memory**

Before you look at the storage types, it’s important to compare flash-based memory to a spinning disk. Flash-based memory can provide huge amounts of IOPS. However, flash memory and SSDs have unique performance characteristics that must be considered when adopting them in your design.

Flash-based memory systems are in many respects similar to RAM. However, flash is nonvolatile, so data is not subject to loss in the event of a power cycle or similar event. The most important difference between flash and RAM is that RAM can be erased and written almost an infinite number of times. This is referred to as the program/erase (P/E) cycle. Flash has a much smaller P/E count, typically of around 100,000 cycles. Simply, flash memory will wear out.

NAND-type flash is the most common type of memory used in SSD disks, and comes in two versions: single-level cell (SLC) and multi-level cell (MLC). SLC stores 1 bit per internal cell, has a higher P/E cycle count, and is faster and typically more expensive than MLC. MLC stores multiple bits per internal cell, has fewer erase cycles, and consumes more power than SLC—but MLC is cheaper.

In general, NAND flash is organized into 4,096-byte pages, which can be exposed as eight 512-byte sectors or a single 4,096-byte sector. These pages are grouped into blocks of 64 to 1,024 pages, with thousands of blocks per chip. This layout is much like a traditional magnetic disk. It's important to consider that the block is the smallest erasable unit, and when modifying a single page in the block, the entire block is erased and then rewritten, including the modifications.

The net effect of this behavior is that writing to an empty block is very fast. However, if no empty block is available, the flash memory controller must perform the following actions:

1. Read the entire block into the controller’s internal RAM
2. Erase the block in the flash memory
3. Update the block in RAM with the new contents
4. Write the entire block to the flash memory

What starts out as a small write-to-a-sector can turn into a write-to-the-entire-block. This effect is called write amplification. This process also highlights a performance consideration of SSD disk and flash memory: as more memory capacity is consumed on the device, there are fewer empty blocks. Fewer empty blocks mean the memory controller must rewrite more blocks using the actions listed above, resulting in slower performance. Therefore, as SSDs fill up, they run slower.

The flash controller implements a technique called wear-leveling to spread the erase operations across all the blocks of flash in order to extend the useful P/E life. This is used in conjunction with over provisioning, sometimes referred to as OP. OP apportions some of the capacity to be used in wear-leveling, garbage collection, and other controller operations to reduce the write amplification overhead. However, although it should be considered in any design, flash longevity isn’t really the issue it was in the past, with a current flash-based SSD disk carrying very similar lifetime expectations as a magnetic disk.

**Local HDD**

Before you look at SSDs and flash, consider regular spinning HDDs in a DAS architecture. You can leverage enterprise-class server drives and RAID controllers to provide high-performance block storage for virtual machines. Capacity and performance requirements are calculated as in a shared storage design.

**Local SSD**

As with the local HDD design, this option replaces the spinning disk with an SSD disk. This can provide greater IOPS performance than a magnetic disk, and as such is the most common type of DAS architecture in virtual desktop deployments.
As was covered in Flash-Based Memory, it is important to consider that in some cases a magnetic disk can outperform low-cost SSD drives. Lower end and less-sophisticated flash memory-based SSDs can perform very poorly for small random write I/Os, which are characteristic of Windows. The effect can be compounded by the I/O blender effect. If you embark on an SSD DAS architecture, perform the appropriate load simulation at your target steady-state workload to ensure that the performance requirements are satisfied.

**Local Flash**

A more recent implementation of flash is the use of PCIe attached flash memory. These devices are typically deployed in a PCIe form factor, and benefit from the faster interconnect to the host hypervisor. In general, they can support a much greater level of I/O throughput when compared to SSD, and in some cases very large orders of magnitude greater—millions of IOPS vs. thousands. PCIe-based flash is typically presented as a disk device to the hypervisor through a custom driver implementation, which is installed separately from the hypervisor binaries. The same considerations apply with PCIe-based flash as SSD; but some operational considerations differ. Upgrades, maintenance, and swap-out operations are more disruptive, as you must physically remove or replace the device from the server host. This will normally involve a power down. You can swap out an SSD disk with the host running, and when using a supported controller and resilient disk RAID configuration—often without disruption to the host, other than degraded service while the RAID is reconstructed.

**Shared Storage Considerations**

The following options are specific to shared storage implementations:

**Virtual Storage Appliances (VSAs)**

VSAs are an interesting deployment option in that they often augment a virtual desktop storage design by bringing further features and performance in a layer above shared and direct-attached storage architecture.

VSA devices often work with both a DAS and shared storage infrastructure, and can add features such as acceleration, deduplication, and replication. A VSA device can present DAS or shared storage from the hypervisor as iSCSI or NFS targets to the hypervisor, in some cases providing a datastore that is looped back to the same host.

The most common deployment use case is where existing shared storage is in place, but does not provide the performance or availability that is required in the design. More advanced virtual storage appliances can use available hypervisor RAM and present it as a datastore for hosting virtual desktops, providing extremely high levels of I/O performance.

Often VSAs can be deployed on virtual desktop hosts or in a dedicated cluster. This provides replication and high-availability features from commodity shared or DAS storage, which would otherwise not provide these features. One caveat to consider concerns VAAI. VMware does not support or certify VAAI on any VSA appliances, although some vendors advertise these capabilities.

**Storage Area Networks (SAN)**

A storage area network, or SAN, is the original deployment architecture, and one that most IT architects will be the most comfortable and familiar with. With enterprise architectures, often a SAN is a datacenter resource that is common across physical and virtual infrastructure.

The endless permutations of SAN design available today are beyond the scope of this document. However, you must consider the storage protocol and back-end I/O requirements carefully, to ensure that the right levels of performance and capacity are available for steady-state operations.

Almost all top-tier storage vendors today will be able to augment their existing magnetic disk arrays with caching technologies to assist in the performance requirements associated with your virtual desktop design. The assist mechanisms available vary from front-end cache to in-line SLC memory, with automatic tiering or placement of hot data.
You should also consider array-based deduplication for a virtual desktop design, as you are dealing with multiple instances of essentially the same data with respect to your virtual desktop clones. Dedupe, whether in flight as the data is written, or as a background or batch-based process, can significantly reduce capacity requirements for virtual machine storage.

**Hybrid Storage Arrays**
Along with the evolution of classic SAN technology, the storage landscape has seen multiple new vendors provide hybrid storage platforms that cater specifically to virtual desktop or high, sustained IOPS workloads. These hybrid arrays often use a blend of flash and magnetic disk, along with proprietary file and block management at the controller level, to provide automatic tiering and high-performance characteristics.

When compared to a traditional approach with SAN and magnetic disk trays, these hybrid arrays can often achieve the same or better performance via the tiering and caching mechanisms described above, in a smaller footprint of controller and disk. The tradeoff with hybrid arrays is typically in the operational side of the infrastructure. Consider carefully the operation and management of another storage citizen in the datacenter, which might require alternate sizing and scaling methodology alongside traditional enterprise storage.

**Flash Arrays**
Flash-based arrays can truly offer the best of both worlds in providing a standards-based interconnect, enabling all the high-availability features of the hypervisor while providing exceptional performance to the workloads hosted on them.

As discussed at the start of this document, flash-based arrays represent a real shift in the design and operation of virtual infrastructures. The performance requirements of virtual desktop and other Tier-1 application workloads need extensive design and validation to accommodate. The game-changing aspect of a flash-based SAN is that you can begin to stack workloads on this platform without real concern for performance degradation, as the flash array can provide enormous levels of IOPS.

While not quite reaching the ideal of being able to simply deploy and forget, these technologies are altering the workload virtualization landscape. Cost is generally a factor with flash array-based designs. However, this higher cost of acquisition is certain to decrease over time as the technology becomes more prevalent.

**Converged Appliance**
In parallel with the accelerated improvements in shared storage architecture, the IT industry has also witnessed the emergence of the converged appliance device. A converged appliance aggregates the compute, storage, and networking elements of an architecture, and combines them in a pre-sized, pre-validated form factor.

Typically a converged appliance will have at least one (and often more) compute node, plus shared storage and networking all within the chassis, shared among the compute nodes. The networking stack is often 10Gbe or InfiniBand to provide a high-bandwidth, high-capacity interconnection among nodes and to the outside infrastructure. The storage subsystem might comprise RAM, PCIe flash, SSD, and magnetic disk. Automatic caching, tiering, and staging of I/O through the appliance occurs via the proprietary drivers and software integrated with the hypervisor.

An appliance-based approach hugely simplifies the entire design process for an architect. Rather than breaking up a large problem into its component parts and solving them individually, an appliance is a scale-out building block for your virtual desktop infrastructure. Sizing is very straightforward, as appliances are generally validated to provide X IOPS for X users in a single block. Divide the number of users by X to find out how many blocks you need, with a few spares. Or, start at a smaller number and scale out as you prove the concept.

The advantages of this approach must be weighed against the operational factor of deploying such an architecture. As the appliance block approach is essentially a net-new investment in infrastructure, it may require a different operational practice alongside your business-as-usual platforms in the datacenter.

Often the operational overhead is acceptable, given the simplicity and high performance of these architectures. You should consider the consequences carefully, however. Appliance-based deployments are very popular, for all the reasons above, and provide a low barrier to entry in the design and deployment of a virtual infrastructure.
Converged Block Architecture

Block architectures actually predate hybrid arrays in the overall storage landscape, and they share some similarities with a converged appliance. However, there are some differences. A block architecture is comprised of separate integrated compute, networking, and storage components, rather than all in the same chassis as in the converged appliance use case. These converged blocks are constructed in the same rack or racks and sized in accordance to a particular workload, for example virtual desktops.

This is an important deployment consideration: where DAS hosts, arrays, and appliances are installed in the server rack, a block architecture typically is the rack. Block-based architectures for virtual desktop are typically sized for a larger number of initial users when compared to appliances or DAS architectures, given the enterprise nature of the components they are assembled from.

Blocks have evolved in step with other options to include array-based acceleration, caching, and dedupe technologies. They typically incorporate Tier-1 shared storage arrays in their block architecture, so they share many of the features found in component arrays. This is also an important consideration with regard to flexibility in configuration. With a block, you can swap the components you require in the design in and out without drastically affecting the overall sizing of the block.

Block components are also validated together, and often the vendor will pre-assemble and integrate all the components before shipping the complete solution, racked and stacked, ready to be plugged in at the datacenter.

Consider the flexibility and maturity of these configurations when looking at an all-in-one approach in your design. As usual, the operational aspects must be considered alongside the advantages of any approach.
Horizon View 5.2 Storage Enhancements and I/O Storms Revisited

This section will look at the standard features that you can leverage when deploying virtual desktop infrastructure with Horizon View 5.2.

View Storage Accelerator (VSA)

In the Performance and Capacity Considerations section, this paper discussed steady and storm conditions with respect to I/O for our storage subsystem. When you are dealing with storm conditions, the most typical example is a virtual machine boot storm. In this example, you power up 1000 virtual machines, which generates many times greater requests for IOPS than in steady state.

View Storage Accelerator is intended to absorb this read I/O at the host level, and therefore reduce the IOPS requirement at the DAS or shared storage array level. This is implemented as an in-memory cache of common blocks. It is applicable to stateless (floating) as well as persistent (dedicated) desktops, and is completely transparent to the guest virtual machine. During peak events such as boot and login, which are very read intensive, the performance improvement is measured as a net reduction in IOPS to a centralized shared storage array.

Figure 10 shows the effect this has on a 1000-virtual machine boot storm to a shared storage array. The illustration shows ~13,000 IOPS without VSA enabled, and ~3,600 with the feature turned on, a reduction of approximately 70 percent.

![Figure 10: VSA Reduces Net IOPS During Peak Events](image)

The benefits of VSA during boot or login storms translate to a savings you can realize in your storage design. Check Appendix A for more information.
Space-Efficient Sparse Virtual Disks (SE Sparse)

In the Performance and Capacity Considerations section, this document looked specifically at the components of a linked-clone virtual desktop virtual machine, and noted that you must account for a certain percentage of growth per virtual machine. This growth occurs over time, as more unique writes occur on the linked-clone virtual machine.

The SE sparse disk feature was introduced in VMware vSphere 5.1. When you enable it, SE Sparse gives you the ability to reclaim previously used space within the guest OS. If your desktop pools are using linked clones, deploying this can reclaim disk space, and occupy only as much space as is currently being used—helping you combat linked clone growth.

There are two steps involved in the space reclamation feature. The first step is the wipe operation, which frees a contiguous area of space in the virtual machine disk. The second step is the shrink, which unmaps or truncates that area of free space to enable the physical storage to return to the free pool, as shown in Figure 11.

SE sparse disks have a new configurable block allocation size, which you can tune to the recommendations of the storage arrays vendor. Consider SE Sparse when leveraging Horizon View linked clone pools to control linked clone growth.

Tiered Storage

Horizon View has featured the option for tiering linked-clone storage since version 4.5. This document briefly covered an aspect of it in the Performance and Capacity Considerations section. Tiered storage built into Horizon View allows the administrator to select different datastores for the replica and linked clone disk placement at the time you create a desktop pool.

An example of this is to place the replica virtual machines into a very high-performance, low-capacity datastore, and the linked clones into a lesser-performing but higher-capacity datastore, effectively splitting the read/write I/O across the two datastores. You can use tiered storage effectively as part of the overall storage design.
Summary

In summary, there are several perils an architect can easily encounter when embarking on a virtual desktop storage design. The most common issue is underestimating performance requirements when scaling, including capacity and sizing requirements. Proper consideration should be given to virtual machine overhead associated with the hypervisor and 3D features specific to Horizon View.

This document has shown the behavior of your target workload, and the steps you can take in optimizing this; the effect of consolidating multiple instances onto your hypervisor; and obvious and not-so-obvious capacity requirements for your desktop virtual machines. Finally, the paper appraised the high-level technology solutions available to the architect when designing their solution.

It is important to consider these elements and design with appropriate balance for performance, capacity, operational simplicity, and future requirements. The virtual desktop landscape is maturing—this document would have much less content in it if it were written 24 months ago, which is testament to the ferocious pace of innovation in this particular sector of enterprise IT. Vendors and partners are converging on a solution—not just for the virtual desktop, but for mixed high-performance workloads in the datacenter.

Already this means that the mental effort associated with such endeavors is greatly reduced, and there’s hope that many of the steps in the design process will eventually be unnecessary.

A huge amount of very detailed reference material exists from both VMware and the expert community related to the storage considerations of a virtual desktop infrastructure. Following completion of this paper, the reader is advised to continue their research, starting with the References section.

About the Author

Rory Clements, Solutions Architect in End-User Computing at VMware wrote this document.

To comment on this paper, contact the VMware End-User Computing Solutions Management and Technical Marketing team at twitter.com/vmwareeucsmtm.

Acknowledgements

VMware would like to acknowledge the following individuals for their contributions to this document:

Mason Uyeda, Cynthia Hsieh, Duncan Epping, Cormac Hogan, Fred Schimscheimer, Tristan Todd, Manrat Chobchuen, and Marilyn Basanta.
References

Default cluster size for NTFS, FAT, and exFAT – Windows NTFS cluster reference

Windows Internals Book – 6th edition

Server and Storage Sizing Guide for Windows 7

Understanding Memory Resource Management in VMware vSphere 5.0

VMX Swap Files

Overhead Memory on Virtual Machines

Disk Defragmentation – Background and Engineering the Windows 7 Improvements – Engineering Windows 7 blog post

IOps? – Duncan Epping, Yellow Bricks blog post

Should I defrag my Guest OS? – Cormac Hogan, VMware vSphere Blog post

Virtual Machine I/O – Fairness versus Performance – Cormac Hogan, VMware vSphere Blog post

An Introduction to Flash Technology – Cormac Hogan, Storage & Virtualization blog post

vSphere 5.1 Storage Enhancements – Part 2: SE Sparse Disks – Cormac Hogan, Storage & Virtualization blog post

VMware View 5.0 3D Reverse Engineered – Andre Leibovici, myvirtualcloud.net blog post

Best Practices for Running vSphere on NFS Storage

VMware vSphere Storage APIs – Array Integration (VAAI)

VMware vSphere 5.1 vMotion Architecture, Performance and Best Practices

View Storage Accelerator – in Practice – Narasimha Krishakumar, VMware End-User Computing Blog post

Storage Protocol Comparison White Paper
Appendix A: VMware Horizon View 500-User Storage Sizing Example

This section works through a simple sizing example, following the methodology outlined in the main document for calculating virtual machine performance and capacity requirements in an example scenario. Compute calculations are out of scope, so you won’t be making any assumptions on that aspect.

Virtual Machine Configuration

This example uses the virtual machine configuration in Table 6.

<table>
<thead>
<tr>
<th>VCPU</th>
<th>VRAM (MB)</th>
<th>VIDEO MEMORY (MB)</th>
<th>OS DISK SIZE (GB)</th>
<th>TARGET IOPS</th>
<th>READ/WRITE IOPS %</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2048</td>
<td>128</td>
<td>24</td>
<td>25</td>
<td>40/60</td>
</tr>
</tbody>
</table>

Table 6: Virtual Machine Configuration

The example uses a Windows 7 32-bit deployment.

Storage Technology Selection and RAID Type

For simplicity, assume a standard shared storage array with a combination of magnetic disk and some caching and acceleration features, presenting block-level storage to your VDI hosts. This will be presented over a FC fabric interconnect to the VDI hosts. All virtual machine files will be on the storage array.

In order to correctly size the IOPS requirements, you must choose a RAID level for your LUNs in advance. This example uses RAID-5.

Virtual Machine Clone Type

As this design is for Horizon View 5.2, elect to leverage View Composer linked clones in the deployment of your 500 virtual machine pool.

Performance Sizing Calculation

Following the guidance laid out earlier, break down the performance requirements as follows:

\[(\text{Target IOPS} \times \text{Read I/O %}) + ((\text{Target IOPS} \times \text{Write I/O %}) \times \text{RAID Penalty}) = \text{IOPS}\]

Your target IOPS are \(25 \times 500 = 12,500\) IOPS for steady-state operations.

Therefore,

\[(12,500 \times 0.4) + ((12,500 \times 0.6) \times 4) = \text{IOPS}\]

Which is:

\[(5,000) + (30,000) = 35,000\] IOPS

This is the total back-end requirement of 35,000 IOPS for your 500 virtual machines in steady state.
Capacity Sizing and LUN Allocation

LUN Allocation and Performance per LUN

Following VMware best practice, assign no more than 64 linked clone virtual machines per LUN. You should require \( \frac{500}{64} \) number of LUNs, which is 7.8, an unusable number—round it up to 8.

Take 35,000 IOPS and divide by the number of LUNs. This shows that each LUN must support 4,375 IOPS, which will feed into your disk allocation for your proposed LUN layout.

You should also provision for storm conditions. Allowing an extra 20 percent of the overall IOPS should satisfy that requirement.

For sizing, remember that your linked clones will break down as follows:

\[
\text{Replica} + \text{Linked Clone} + \text{Growth\%} + \text{Size of .VSWP} + \text{Overhead} = \text{Storage per Virtual Machine}
\]

Replica Overhead

As a best practice, allow for 2 times the replica size on each LUN you are going to provision. You can already calculate that there will be a 48GB requirement on each LUN: \( 48 \times 8 = 384 \text{GB} \) for replicas.

Linked Clone Base Size

Assume a base linked clone size of 50 percent of the replica, which gives you 12GB per linked clone virtual machine.

Virtual Machine SWAP

Reserve 50 percent of the allocated vRAM in this example. Assume a 1024MB swap file for each linked clone plus the overhead, which is 260MB; therefore you have a 1284MB SWAP requirement for each linked clone. Check the References section for the vSphere documentation on sizing swap and overhead requirements.

Linked Clone Growth

For this example, provision for 20 percent linked clone growth: \( 12 \times 0.2 = 2.4 \text{GB} \) per linked clone.

Table 7 shows initial capacity requirements for a single virtual machine.

<table>
<thead>
<tr>
<th>VIRTUAL MACHINES</th>
<th>REPLICA (MB)</th>
<th>LINKED CLONES (MB)</th>
<th>SWAP (MB)</th>
<th>GROWTH (MB)</th>
<th>TOTAL (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>12288</td>
<td>1284</td>
<td>2458</td>
<td>16030</td>
</tr>
</tbody>
</table>

Table 7: Initial Capacity Requirements

Table 8 shows requirements scaled up for 500.

<table>
<thead>
<tr>
<th>VIRTUAL MACHINES</th>
<th>REPLICA (GB)</th>
<th>LINKED CLONES (GB)</th>
<th>SWAP (GB)</th>
<th>GROWTH (GB)</th>
<th>TOTAL (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>384</td>
<td>6000</td>
<td>628</td>
<td>1200</td>
<td>8212</td>
</tr>
</tbody>
</table>

Table 8: Initial Capacity Requirements Scaled Up for 500

You need about 8.2TB of storage capacity to support 500 users. That number divides nicely into your proposed number of LUNs, so you can provision approximately 1.1TB per LUN to satisfy your capacity requirements.
Horizon View Features

For your design, enable View Storage Accelerator and provision the maximum of 2048MB per VDI host to act as cache. Create your virtual machine with the SE sparse virtual disk format.

Sizing Summary

To summarize, your storage design follows the LUN layout in Table 9.

<table>
<thead>
<tr>
<th>LUN #</th>
<th>SIZE (TB)</th>
<th>TARGET STEADY-STATE IOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>2</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>3</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>4</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>5</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>6</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>7</td>
<td>1.1</td>
<td>4,375</td>
</tr>
<tr>
<td>8</td>
<td>1.1</td>
<td>4,375</td>
</tr>
</tbody>
</table>

Table 9: Storage Design Example

With respect to the FC fabric, your multipathing design should include provisions for resilience and performance requirements for the layout. You must ensure that the transmission paths are resilient and balanced appropriately, with contingency for a path to fail and continue to accommodate your steady-state load.