Module 6: Backing Azure Solutions with Azure Storage

Contents:

Module overview

This module describes how many Azure services use the Azure Storage service as a backing store for other application solution in Azure. The module dives into critical considerations when using Azure Storage as a supplemental service for an all-up Azure solution.

Objectives

After completing this module, students will be able to:

- Determine the ideal pricing option for Azure Storage based on a solution's requirements.
- Identify performance thresholds for the Azure Storage service.
- Determine the type of Storage blobs to use for specific solution components.
- Use the Azure Files service for SMB operations.
- Identify solutions that could benefit from the use of StorSimple physical or virtual devices.

Lesson 1: Pricing

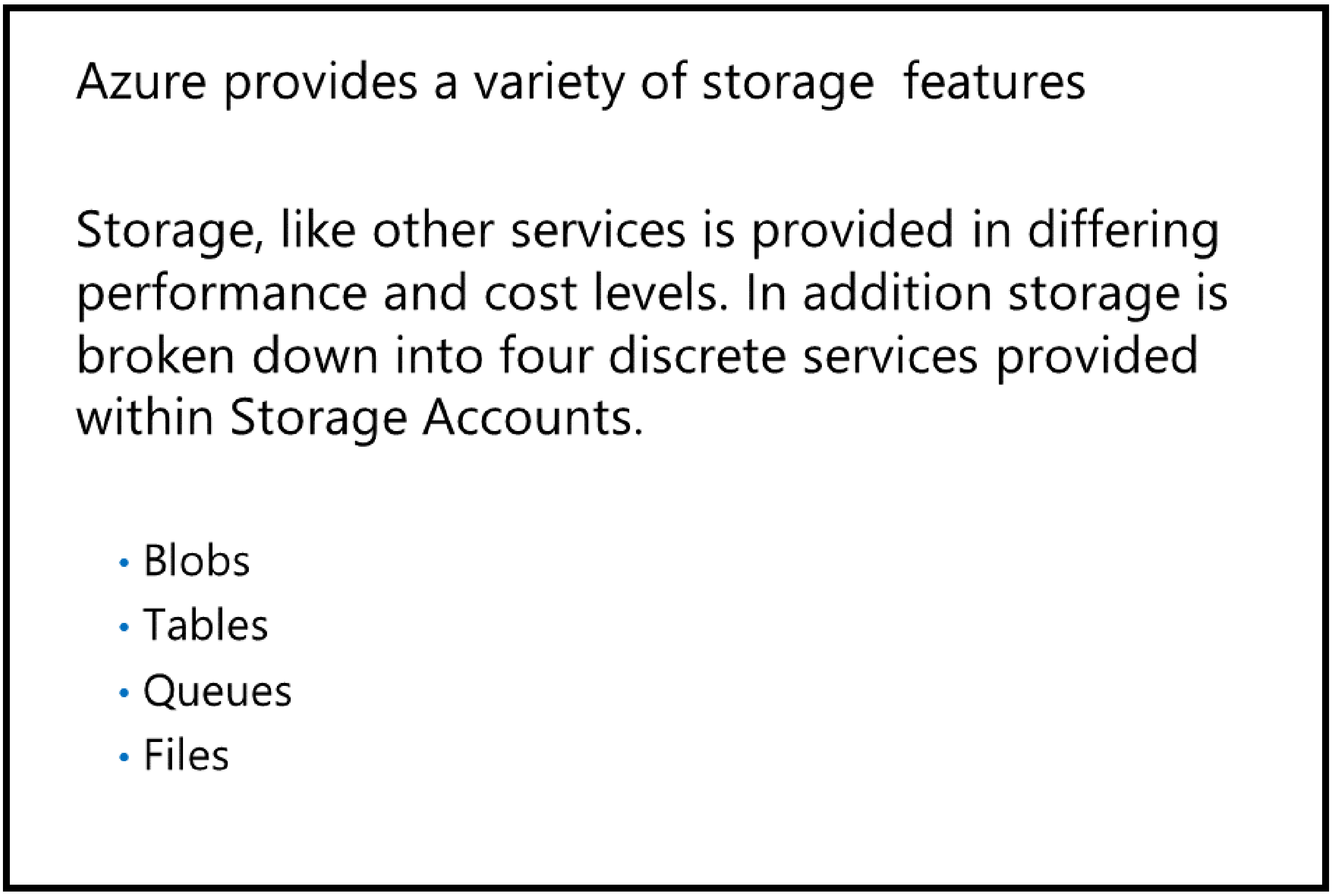
Azure storage has many options which affect performance and pricing. To be able to select the best storage for your infrastructure and applications requires a knowledge of what to store, how to store it, access is and pay for it. This lesson covers the essential elements of Azure Storage.

Lesson objectives

After completing this lesson, you will be able to:

- Describe the three Azure storage accounts and features
- Understand the redundancy options available in Azure Storage
- Chose which Azure storage account works best for your requirements
- Understand how to access, share and delegate storage accounts

Azure Storage



Azure provides a variety of storage features

Storage, like other services is provided in differing performance and cost levels. In addition storage is broken down into four discrete services provided within Storage Accounts.

- Blobs
- Tables
- Queues
- Files

Azure Storage is massively scalable, so you can store and process hundreds of terabytes of data to support the big data scenarios required by scientific, financial analysis, and media applications. You can also store the small amounts of data required for a small business website as billing is calculated by usage, not capacity. Storage uses an auto-partitioning system that automatically load-balances your data based on traffic. Since storage is elastic and decoupled from your application, you can focus on your workload while relying on Storage's elastic capabilities to scale to meet the demand for your applications.

All Storage services can be accessed using a REST API. Client libraries are also available for popular languages and platforms such as:

- .NET
- Java/Android

- Node.js
- PHP
- Ruby
- Python
- PowerShell

Microsoft Azure provides a storage subsystem that allows users to create a variety of different types of storage. The storage can be provided in differing levels of performance.

Storage accounts are subdivided into available storage services all intended for different functions:

- **Blobs** – VHDs and large blocks of data (images or documents)
- **Tables** – structured data (a NoSQL store)
- **Queues** – for message passing in applications and for backlog processing
- **Files** – Fully managed SMB 3.0 file shares

To store and manage these storage services Azure uses a container called a Storage Account.

This lesson deals with the storage account; a subsequent lesson will deal with managed Disks for Virtual Machine disk storage without the need for a storage account.

Azure Storage Accounts

There are three types of Azure Storage accounts:

Type of Account	General Purpose Standard	General Purpose Premium	Blob Storage (hot and cool access tiers)
Services Supported	Blob, File, Queue services	Blob service	Blob service
Types of Blobs supported	Block blobs, Page blobs and Append blobs	Page blobs	Block blobs and Append blobs

There are also three types of entities in Azure Storage accounts:

Blobs

Blobs are available in 3 forms all intended for different purposes:

- **Page blobs** lend themselves to storing random access files up to 8TB in size, ideal for VHD storage for Virtual Machines

- **Block blobs** are for images or other documents and files up to 4TB in size
- **Append blobs** are similar in format to block blobs but allow append operations and are ideal for auditing and logging applications such as log or monitoring data from many sources.

Tables

“Massive auto-scaling NoSQL store”. Ideal for: User, device and service metadata, structured data. Features include:

- Schema-less entities with strong consistency
- No limits on number of table rows or table size
- Dynamic load balancing of table regions
- Best for Key/value lookups on partition key and row key
- Entity group transactions for atomic batching

Queues

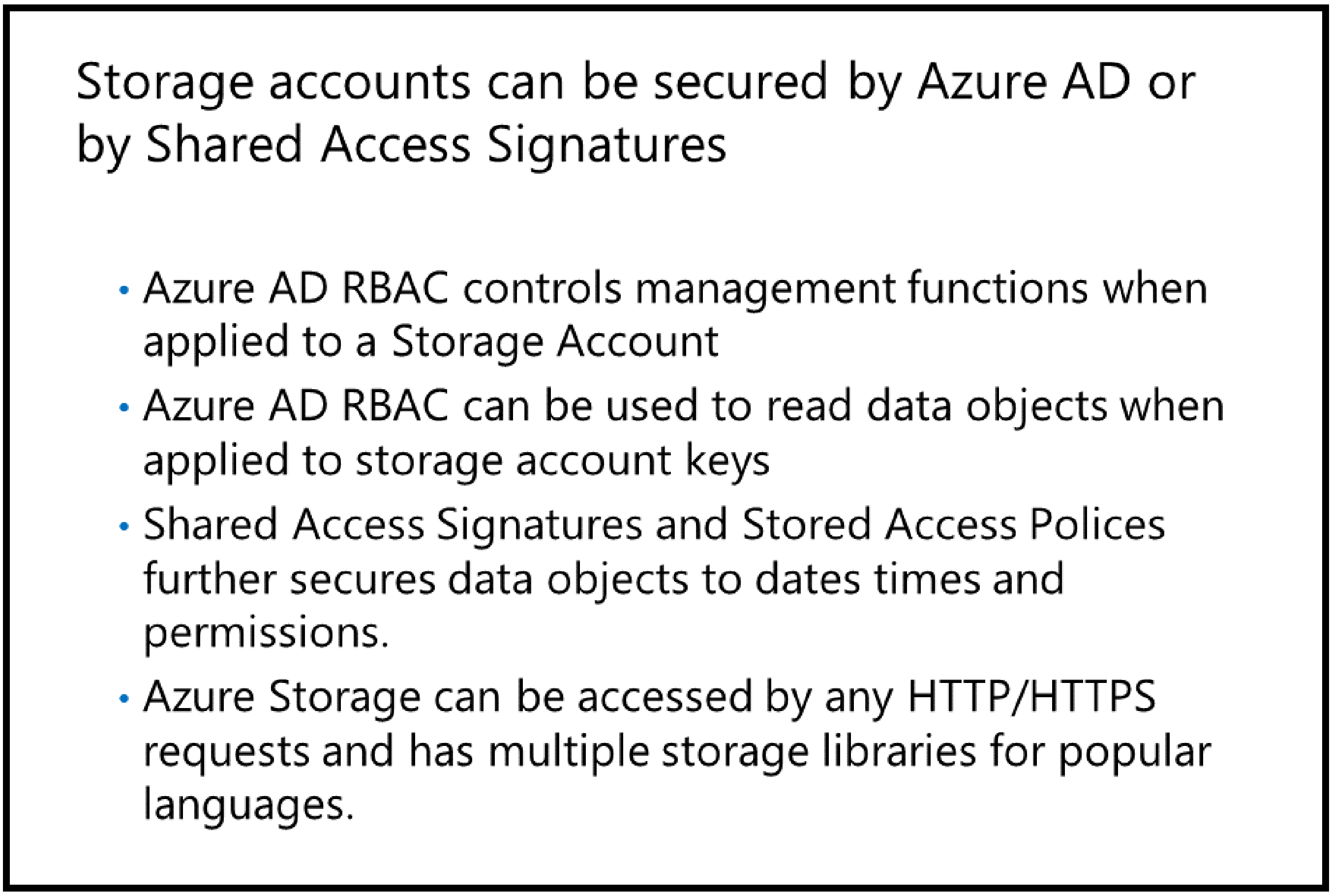
“Reliable messaging system at scale for cloud services”. Ideal for: Data sharing, Big Data, Backups. Functions include:

- Decouple components and scale them independently
- Scheduling of asynchronous tasks
- Building processes/work flows

Features include:

- No limits on number of queues or messages
- Message visibility timeout to protect from component issues U
- UpdateMessage to checkpoint progress part way through

Storage Account Security



Storage accounts can be secured by Azure AD or by Shared Access Signatures

- Azure AD RBAC controls management functions when applied to a Storage Account
- Azure AD RBAC can be used to read data objects when applied to storage account keys
- Shared Access Signatures and Stored Access Polices further secures data objects to dates times and permissions.
- Azure Storage can be accessed by any HTTP/HTTPS requests and has multiple storage libraries for popular languages.

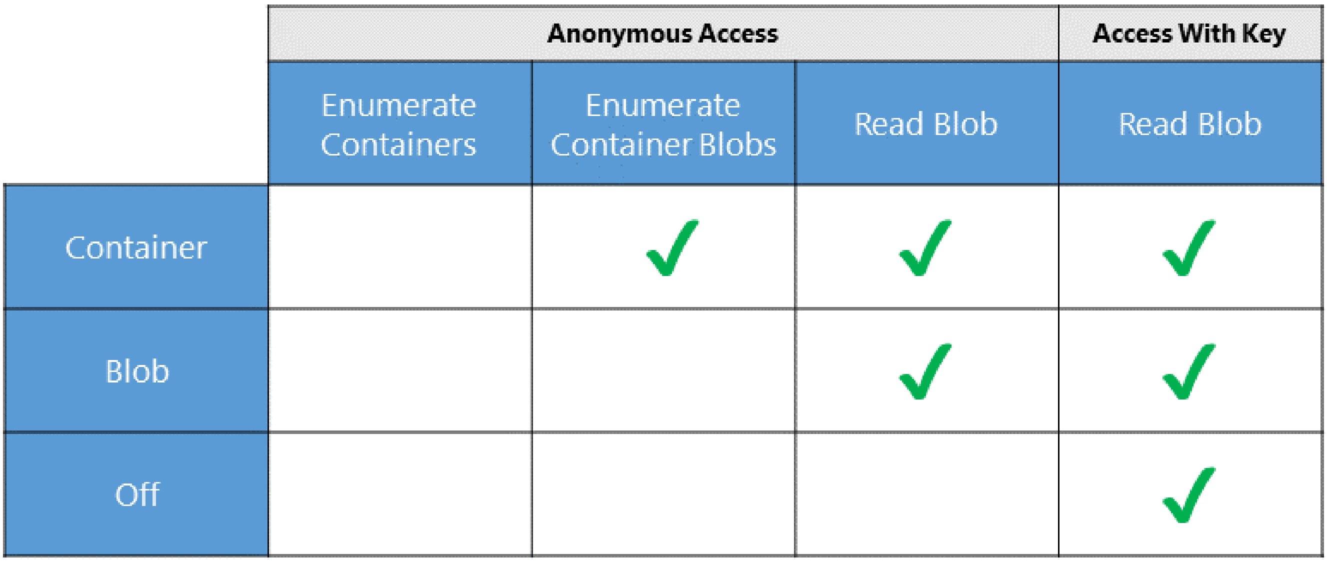
Access to storage accounts is controlled by Storage Account Keys (there are primary and secondary to allow for key refreshes whilst maintaining access). If you have the key, you have access to the account and all the data in the account. Container security is also provided for blob storage at the time of creation; the container can be public, private or read-only.

To provide greater security, Azure provides Azure AD RBAC to allow for management functions on the Storage account and the use of RBAC on the storage account keys to prevent incorrectly applied access to the data itself. There are also Shared Access Signatures and Stored Access Policies to provide more granular access at the container, blob, queue or table level. A shared access policy adds the ability to revoke, expire or extend access. The use of all of these security features is a design decision when building your infrastructure or application.

Container Security

Typically, only the owner of a storage account can access resources within that account. If your service or application needs to make these resources available to other clients, you have various options available. First, you can make the public access key generally available. This is not typically recommended as this key gives individuals full access to your entire storage account and its management operations. Another, more common option is to manage access to the entire container.

This access can be managed using the Public Read Access property of a specific container.



	Anonymous Access			Access With Key
	Enumerate Containers	Enumerate Container Blobs	Read Blob	Read Blob
Container		✓	✓	✓
Blob			✓	✓
Off				✓

FIGURE 6.1: CONTAINER SECURITY OPTIONS

The Public Read Access property controls what data is available anonymously for your container. You can select the following values for the Public Read Access setting:

- **Container**. Blobs in a container can be enumerated. The container metadata is also accessible. Individual blobs within this container and their properties can also be accessed with this setting.
- **Blob**. Only individual blobs and their properties in this container can be accessed. Blobs are not allowed to be enumerated.
- **Off**. With this setting, enumeration of blobs is not allowed. Individual blobs and their properties are also not accessible. You must use your access keys to access any data about this container or its blobs.

Shared Access Signatures

A shared access signature is a URI that grants restricted access rights to containers, blobs, queues, and tables. You can provide a shared access signature to clients who should not be trusted with your storage account key but to whom you wish to delegate access to certain storage account resources. By distributing a shared access signature URI to these clients, you can grant them access to a resource for a specified period of time, with a specified set of permissions.

A shared access signature can grant any of the following operations to a client that possesses the signature:

- Reading and writing page or block blob content, block lists, properties, and metadata
- Deleting, leasing, and creating a snapshot of a blob
- Listing the blobs within a container
- Adding, removing, updating, and deleting queue messages
- Getting queue metadata, including the message count

- Querying, adding, updating, deleting, and upserting table entities
- Copying to a blob from another blob within the same account

The shared access signature URI query parameters incorporate all of the information necessary to grant controlled access to a storage resource. The URI query parameters specify the time interval over which the shared access signature is valid, the permissions that it grants, the resource that is to be made available, and the signature that the storage services should use to authenticate the request.

For example, you may wish to have your client application access a resource (container: pictures, blob: profile.jpg) at this URI:

GET https://[account].blob.core.windows.net/pictures/profile.jpg

You could give the client application your access key to manage your entire subscription or make the pictures container anonymously accessible. Neither of these is ideal solutions, so you instead chose to generate a shared access signature. You simply append the generated signature to the end of your URI like this:

GET https://[account].blob.core.windows.net/pictures/profile.jpg?sv=2012-02-12&st=2009-02-09&se=2009-02-10&sr=c&sp=r&si=YWJjZGVmZw%3d%3d&sig=dD80ihBh5jfNpymO5Hg1IdiJIEvHcJpCMiCMnN%2fRnbI%3d

The signature is using version 2012-02-12 of the storage API; it allows read access is beginning from 02/09/09 to 02/10/09 to the container.

The structure of a SAS URI is the composition of multiple pieces.

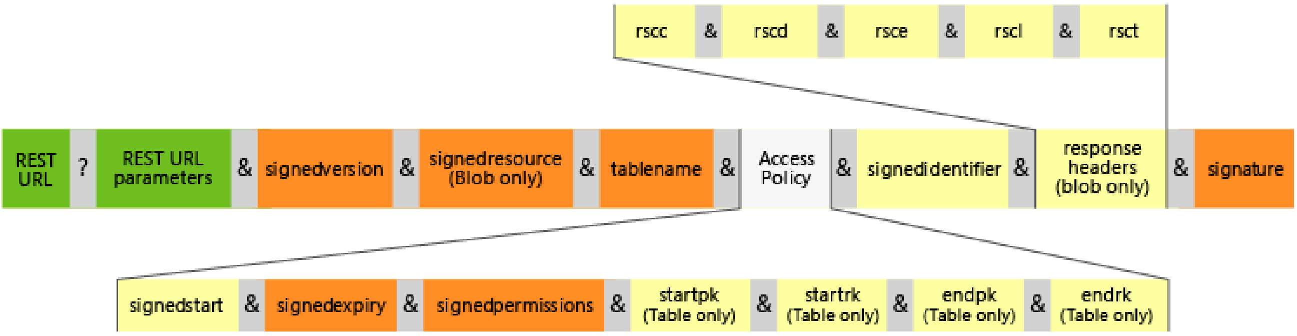


FIGURE 6.2: SAS URI CONSTRUCTION

Now the resource is temporarily accessible by this particular client.

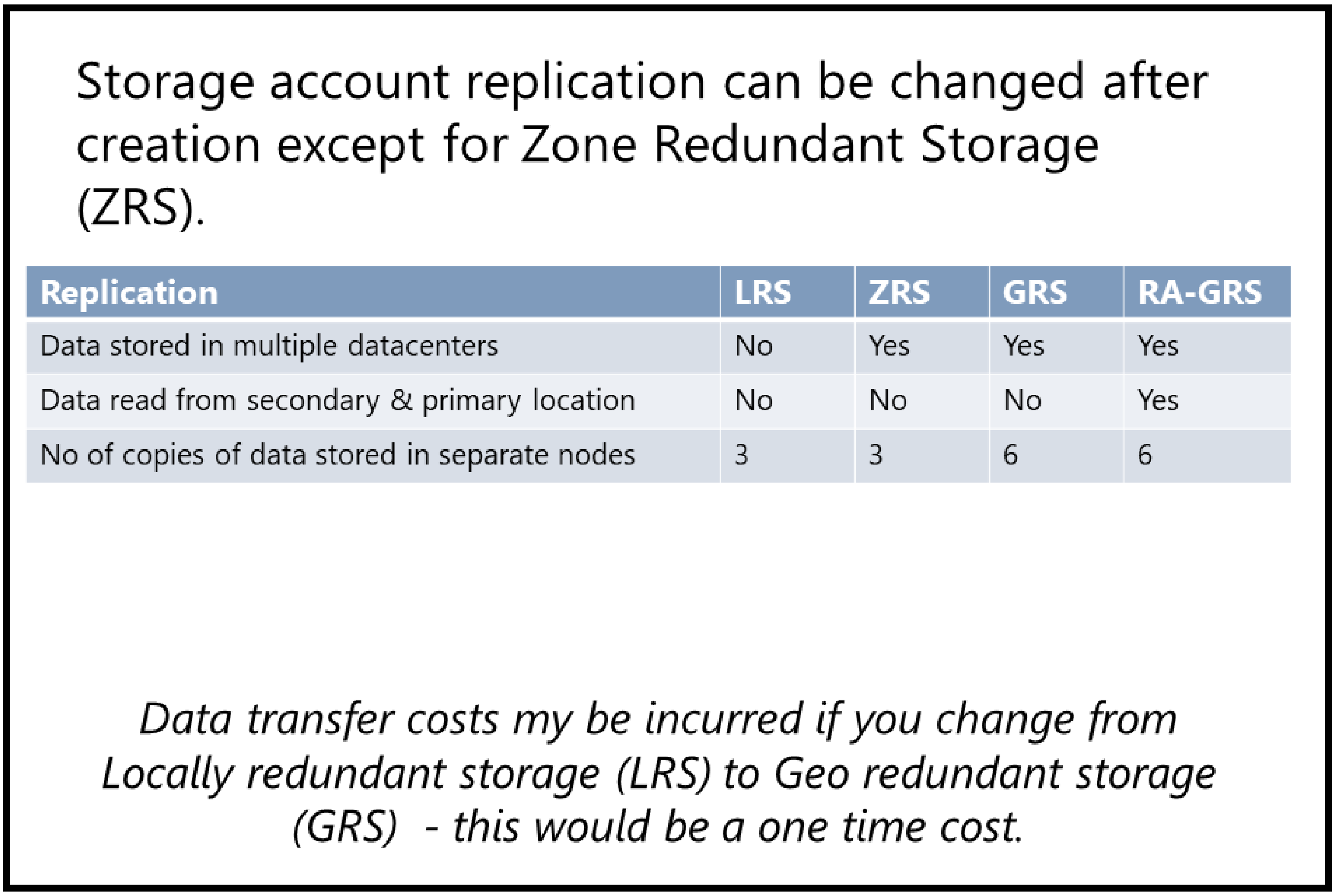
Stored Access Policies

Azure SAS also supports server-stored access policies that can be associated with a specific resource such as a table or blob. This feature provides additional control and flexibility compared to application-generated SAS tokens and should be used whenever possible. Settings defined in a server-stored policy can be changed and are reflected in the token without requiring a new token to be issued, but settings defined in the token itself cannot be changed without issuing a new token.This approach also makes it possible to revoke a valid SAS token before it has expired.

A stored access policy provides an additional level of control over shared access signatures on the server side. Establishing a stored access policy serves to group shared access signatures and to provide additional restrictions for signatures that are bound by the policy. You can use a stored access policy to change the start time, expiry time, or permissions for a signature, or to revoke it after it has been issued.

New shared access signatures are then generated from an existing stored access policy. A maximum of five access policies may be set on a container, table, or queue at any given time. To revoke a stored access policy, you can either delete it or rename it by changing the signed identifier. Changing the signed identifier breaks the associations between any existing signatures and the stored access policy. Deleting or renaming the stored access policy immediately effects all of the shared access signatures associated with it.

Storage Account Replication



Storage account replication can be changed after creation except for Zone Redundant Storage (ZRS).

Replication	LRS	ZRS	GRS	RA-GRS
Data stored in multiple datacenters	No	Yes	Yes	Yes
Data read from secondary & primary location	No	No	No	Yes
No of copies of data stored in separate nodes	3	3	6	6

Data transfer costs my be incurred if you change from Locally redundant storage (LRS) to Geo redundant storage (GRS) - this would be a one time cost.

The data in your Microsoft Azure storage account is always replicated to ensure durability and high availability. At a minimum, your data is stored in triplicate. You may also choose extended replication options for scenarios where you require your data to be replicated across geography.

Options include:

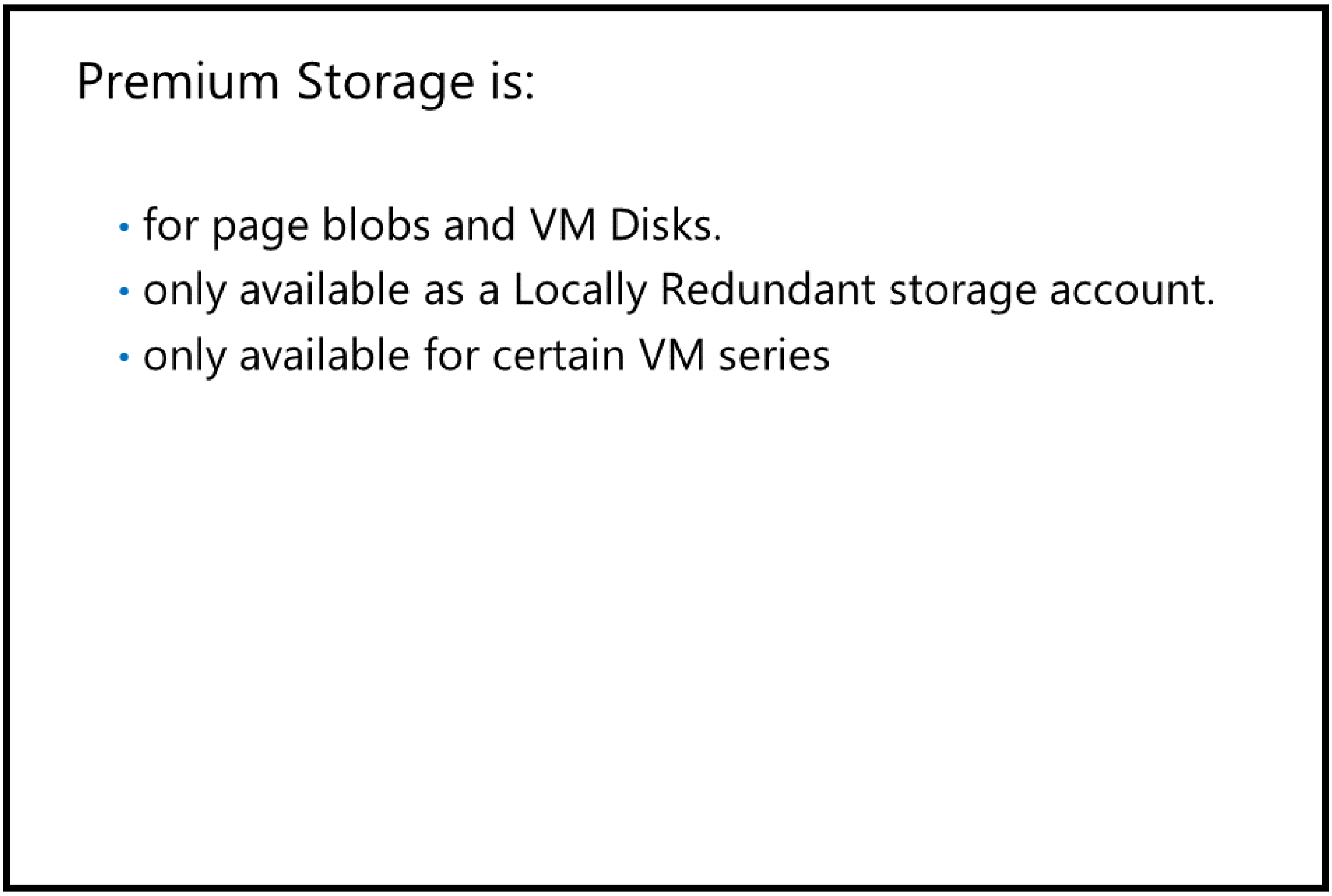
- **Locally redundant storage (LRS)**: Locally redundant storage maintains three copies of your data. LRS is replicated three times within a single facility in a single region. LRS protects your data from normal hardware failures, but not from the failure of a single facility. LRS is the minimum amount of replication.
- **Zone-redundant storage (ZRS)**: Zone-redundant storage maintains three copies of your data. ZRS is replicated three times across two to three facilities, either within a single region or across two regions, providing higher durability than LRS. ZRS ensures that your data is durable within a single region.

Geo-redundant storage (GRS): Geo-redundant storage is enabled for your storage account by default when you create it. GRS maintains six copies of your data. With GRS, your data is replicated three times within the primary region, and is also replicated three times in a secondary region hundreds of miles away from the primary region, providing the highest level of durability. In the event of a failure at the primary region, Azure Storage will failover to the secondary region. GRS ensures that your data is durable in two separate regions.

- **Read access geo-redundant storage (RA-GRS)**: Read access geo-redundant storage replicates your data to a secondary geographic location, and also provides read access to your data in the secondary location. Read-access geo-redundant storage allows you to access your data from either the primary or the secondary location, in the event that one location becomes unavailable. RA-GRS is also used in scenarios where reporting and other read-only functions can easily be distributed to the replica instead of the primary therefore spreading application load across multiple instances.

> **Note:** Geographically distributed replicas receive any replication asynchronously. This means that your replica is eventually consistent and could possibly have older data if you access the replica before the replication operation from the primary is complete.

Storage Performance & Pricing



Premium Storage is:

- for page blobs and VM Disks.
- only available as a Locally Redundant storage account.
- only available for certain VM series

Storage Accounts are available in Standard and Premium tiers. Azure Premium Storage delivers low-latency and high-performance disk support for virtual machines (VMs). Premium Storage stores data on solid-state drives (SSDs). Standard VM disks may be migrated to Premium Storage. Using multiple disks gives your applications up to 256 TB of VM storage. Premium Storage provides up to 80,000 I/O operations per second (IOPS) per VM, and a disk throughput of up to 2,000 megabytes per second (MB/s) per VM.

Premium Storage allows the lift-and-shift of demanding applications such as Dynamics AX, Dynamics CRM, SQL Server, Exchange Server and SharePoint farms to Azure. Applications that require consistent high performance and low latency such as SQL Server, Oracle, MongoDB, MySQL, and Redis can run happily in Azure Premium Storage. Premium Storage accounts can only be created with LRS redundancy. Not all Azure VM sizes can take advantage of Premium disk support. Plan carefully your disk and VM sizes to ensure maximum price and efficiency benefits. Currently Premium Storage is available on DS-series, DSv2-series, GS-series, Ls-series, and Fs-series VMs within Azure. Premium disks are always available for use. The storage is allocated on creation and is charged for the whole disk size rather than the data used. This makes premium disks more expensive.

Lesson 2: Blob Storage

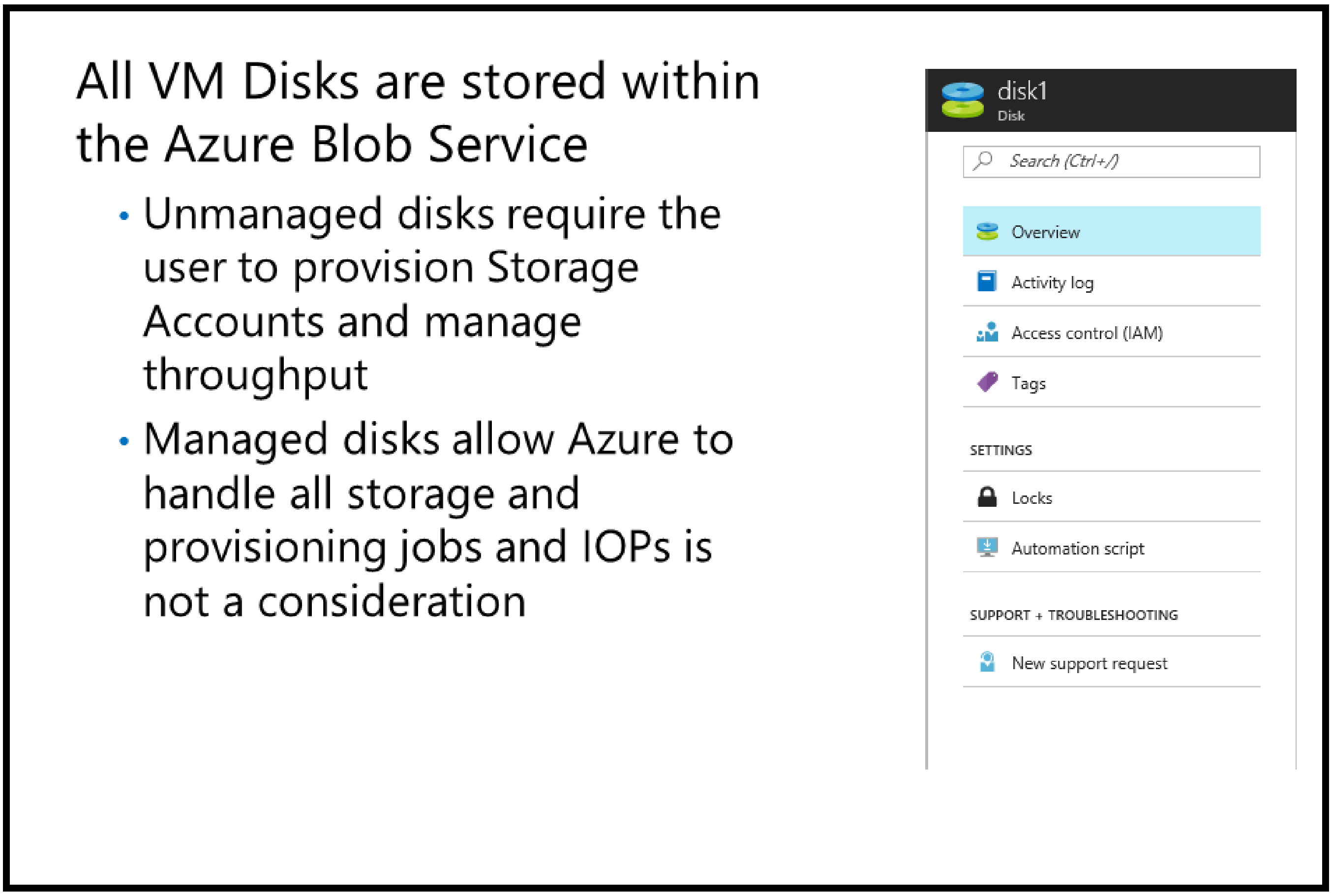
The Blob Storage Service provides the infrastructure for Azure to store and manage the Block and Page blobs required to host Azure VM Disks and all other random-access Blobs as well as the block blob documents, images, and files. This lesson provides insight into the managed and unmanaged disk service as well as the blob service within a storage account.

Lesson objectives

After completing this lesson, you will be able to:

- Describe managed disks, snapshots, and disk exports.
- Decide whether to deploy VM disks as managed or unmanaged.
- Deploy a managed disk.
- Deploy data disks to a VM for a storage space.

Blob Storage



All VM Disks are stored within the Azure Blob Service

- Unmanaged disks require the user to provision Storage Accounts and manage throughput
- Managed disks allow Azure to handle all storage and provisioning jobs and IOPs is not a consideration

Azure Blob storage was introduced in the previous lesson as part of the overall Azure Storage platform. In this lesson, we cover the Azure Blob service from an Azure IaaS VM Disk perspective. Looking at VM disks in a managed and unmanaged form and the premium versus standard performance tier of both.

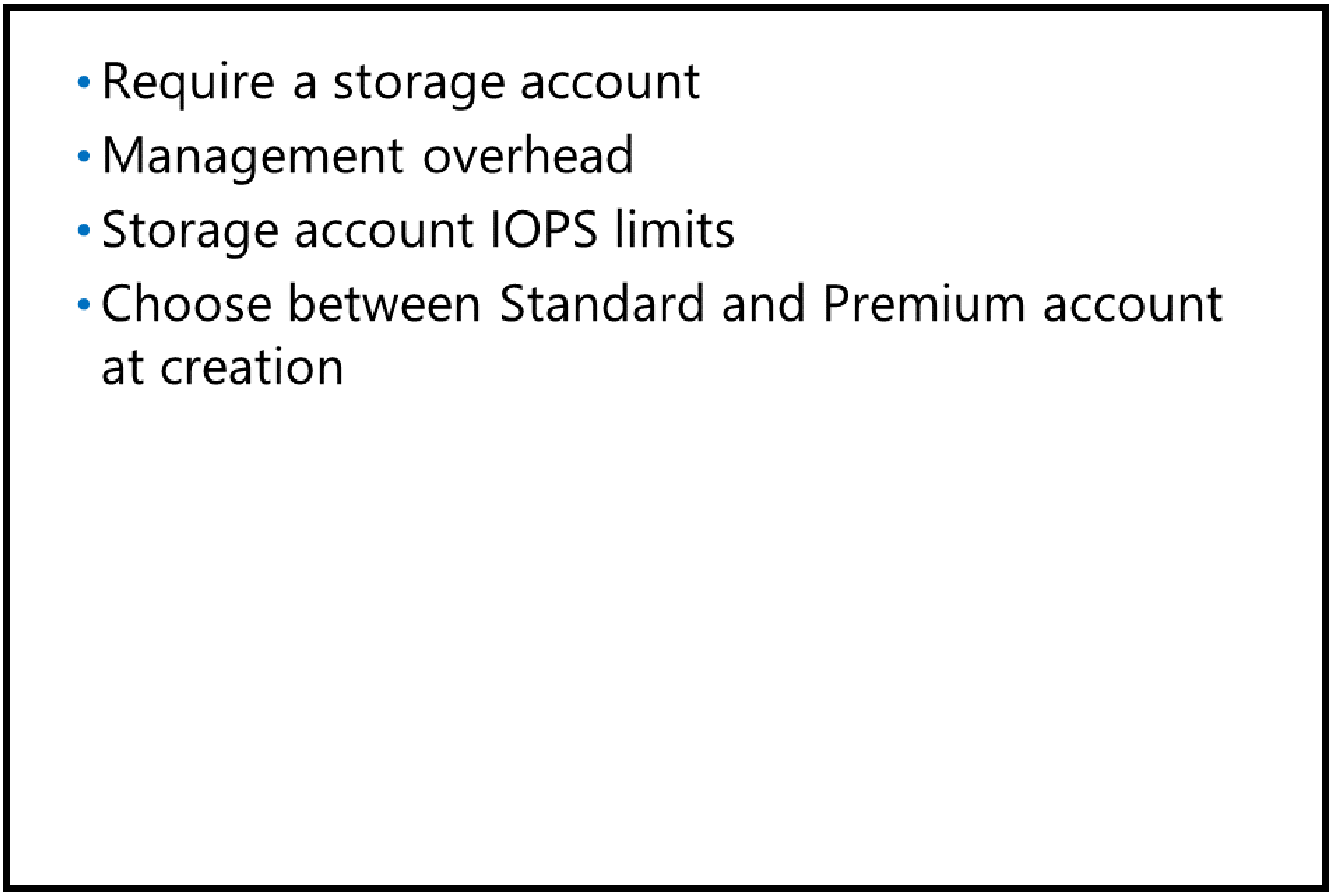
VM Disks

Azure IaaS VMs can attach OS and Data Disks. These disks can be premium storage (SSD based) or Standard storage (HDD). The performance of the disks depends upon the VM in which it is attached and the performance tier in which it is created. The management of VM data disks has been a challenging task in the past due to the performance limitations of a storage account. These were limited to a maximum of 20,000 IOPS per account. The Managed Disk service has provided a new feature in which Azure handles to underlying storage infrastructure and carries out all the redundancy and high availability tasks as well as ensuring that throughput is not throttled by the 20000 IOPS limit in storage accounts. Managed and un-managed disks have different feature sets and performance.

The choice of whether to use Azure Files, Azure Blobs or Azure Disks is one primarily dictated by the data to be stored and the use of that data.

Feature	Description	Suggested Use
Files	SMB 3.0 interface, client libraries and a REST interface access from anywhere to stored files	Application lift and shift using native file system API, Windows File Shares
Blobs	Client libraries, REST interface, for unstructured data stored in Block blobs	Streaming and random access, access application data from anywhere.
Disks	Client libraries, REST interface for persistent data accessed from a VHD	Access to data inside a Virtual machine on a VHD, lift and shift file system API apps that write to persistent disks

Un-Managed Disks



- Require a storage account
- Management overhead
- Storage account IOPS limits
- Choose between Standard and Premium account at creation

Un-managed disks are available in Standard and Premium tiers.

Un-managed disks are available in Standard and Premium tiers. To use an un-managed disk, it is first necessary to create a Storage account and a container within the blob service of that account.

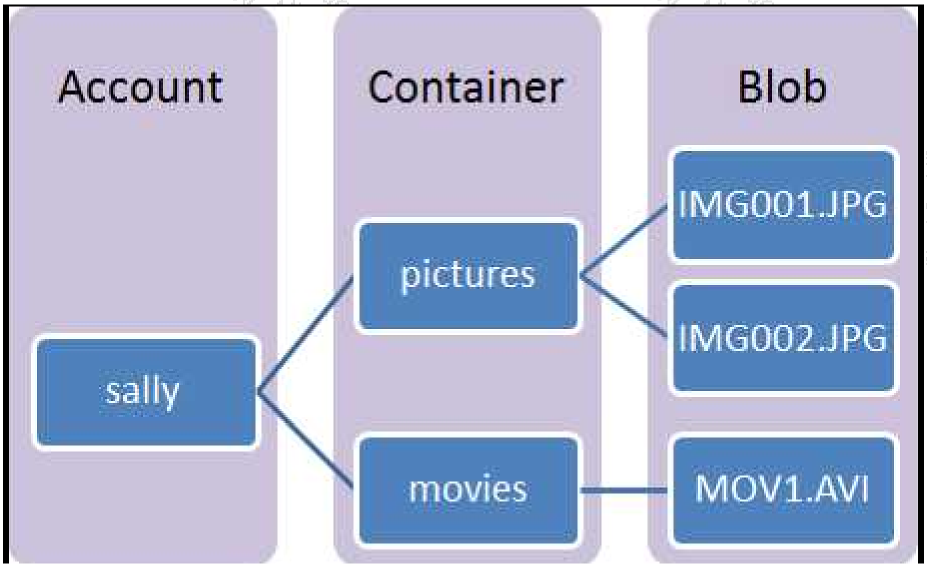
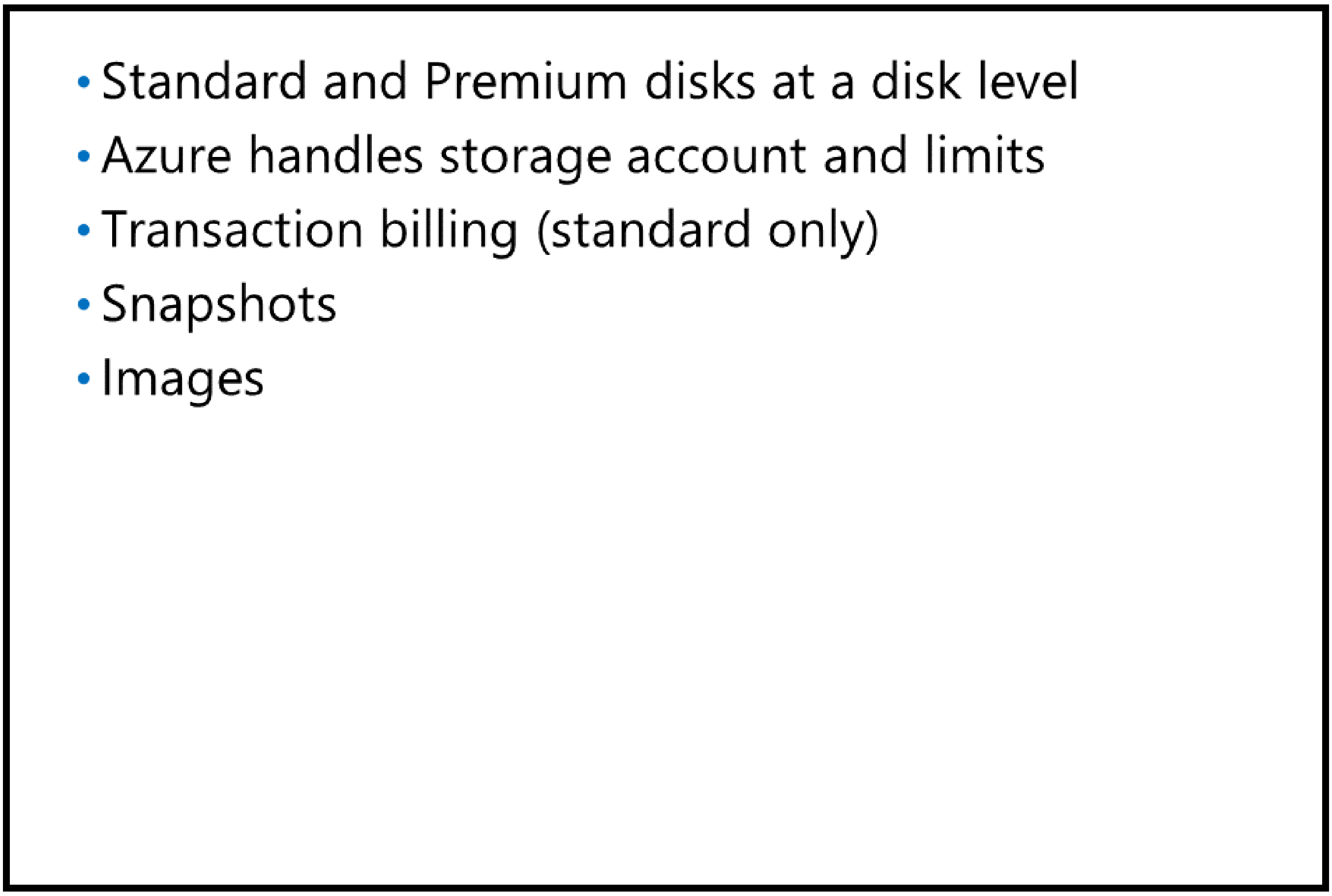


FIGURE 6.3: STORAGE HIERARCHY

Once created blobs (including VM disks) can then be stored in the container. Each storage account has a limit of 20000 IOPS throughput. To be able to provide maximum performance on a striped storage space with multiple disks inside a VM it would be necessary to spread those data disks across many storage accounts.

Any Azure VM size can have multiple standard disks attached. A storage account can only store one type of disk. The performance of the storage account is chosen at time of creation. Un-managed standard disks can have disk snapshots (using Azure CLI or PowerShell), and VM images can be created with disks included.

Managed Disks



- Standard and Premium disks at a disk level
- Azure handles storage account and limits
- Transaction billing (standard only)
- Snapshots
- Images

Azure Managed Disks simplifies the creation and management of Azure IaaS VM Disks. When using the Managed Disk service, Azure manages the storage accounts in which the VM disks are stored. With un-managed disks, you need to create a storage account, a container and decide on Standard or Premium storage for the account. Then you need to ensure the VM disks are spread efficiently around several storage accounts. With Managed disks, the requirement is merely to choose the storage type and the disk size. Azure does the rest.

Managed disks provide several other benefits. First amongst these is the ability to upgrade a standard disk to a premium disk and downgrade a premium disk to a standard disk. Because there is no performance level for the underlying storage account, the Azure Managed disk architecture allows quick and straightforward performance tier changes. The only requirement is to detach the disk from a VM before it is upgraded or downloaded.

The architecture of Managed disks also provides:

- Scalable and straightforward VM deployment – no need to provision additional storage accounts when scaling the number of disks, the ceiling of 20000 IOPS does not exist. In addition, storing custom images on VHD files can now be accomplished with one storage account per region no need for additional storage accounts.
- Managed Disks allows up to 10000 VM disks per subscription. Using managed disks in VM Scale Sets allows up to 1000 VMs per scale set using a Marketplace image.
- Better reliability for Availability Sets – Managed disks use a different storage scale unit (or stamp) when using Availability Sets, this further ensures that all the disks in a VM are stored in the same scale unit, and more importantly each VM's disks will be stored in separate scale units ensuring the application relying on those disks remains active.

- Highly durable and available – managed disks are designed for 99.999% availability. Overall Azure manages a ZERO% annualized failure rate

- Granular access control – Azure AD RBAC allows specific permissions to be assigned to managed disks for one or more users or groups. The granularity allows only read access or to preventing export actions on a disk.

- Sizing and Pricing

  - Premium Managed disks are available from P4 which is 32Gb up to P50 which is 4TB in size.

  - Standard Managed disks are available from S4 which is 32GB again up to S50 which is 4TB in size.

  - You are billed for the number of transactions performed on a standard managed disk. Premium disks do not attract transaction charges.

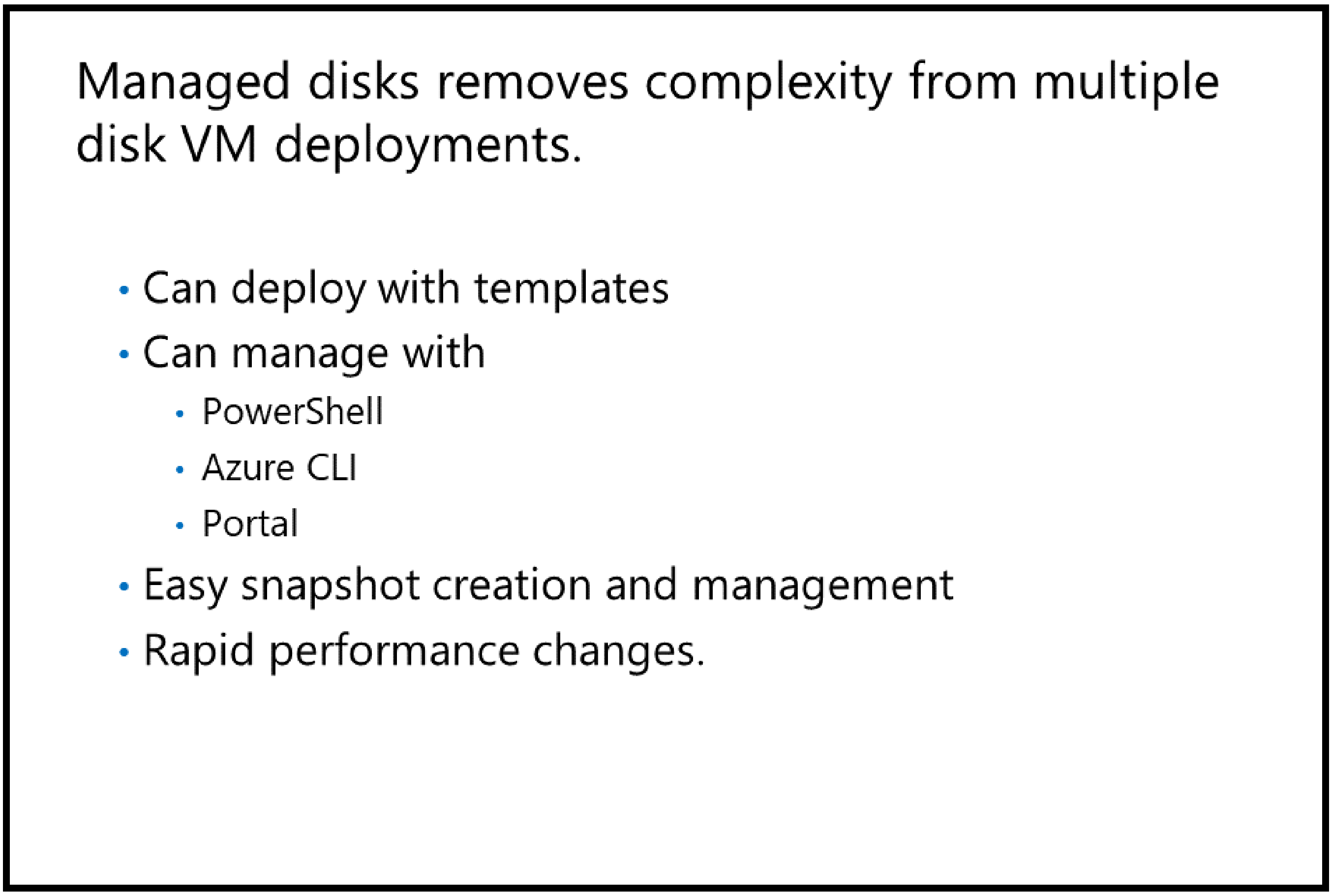
- Managed Disk Snapshots - A Managed Snapshot is a full copy of a managed disk which is stored as a standard read-only managed disk. Snapshots allow backups at any time. These are stand-alone objects and can be used to create new disks. You are only billed for the used size of the data on the disk. So, a 4TB disk that holds 500GB of data the snapshot will be billed for 500GB.

- Images - Managed Disks support creating a managed custom image. This captures in a single image all managed disks attached to a VM.

- Images versus snapshots – An image is a copy of the VM and all disks attached. A snapshot is a single disk copy. For this reason, a snapshot is not a suitable choice for the scenario with multiple striped data disks. No snapshot co-ordination is available.

- Managed Disks and Encryption – By default Managed disks are encrypted by Azure Storage Service Encryption; this provides encryption at rest for disks, snapshots, and images. Azure Disk Encryption is also available at the VM level in Windows this uses BitLocker Drive Encryption. Azure Key Vault integration is included which allows users to bring their own disk encryption keys.

Deployment Considerations



Managed disks removes complexity from multiple disk VM deployments.

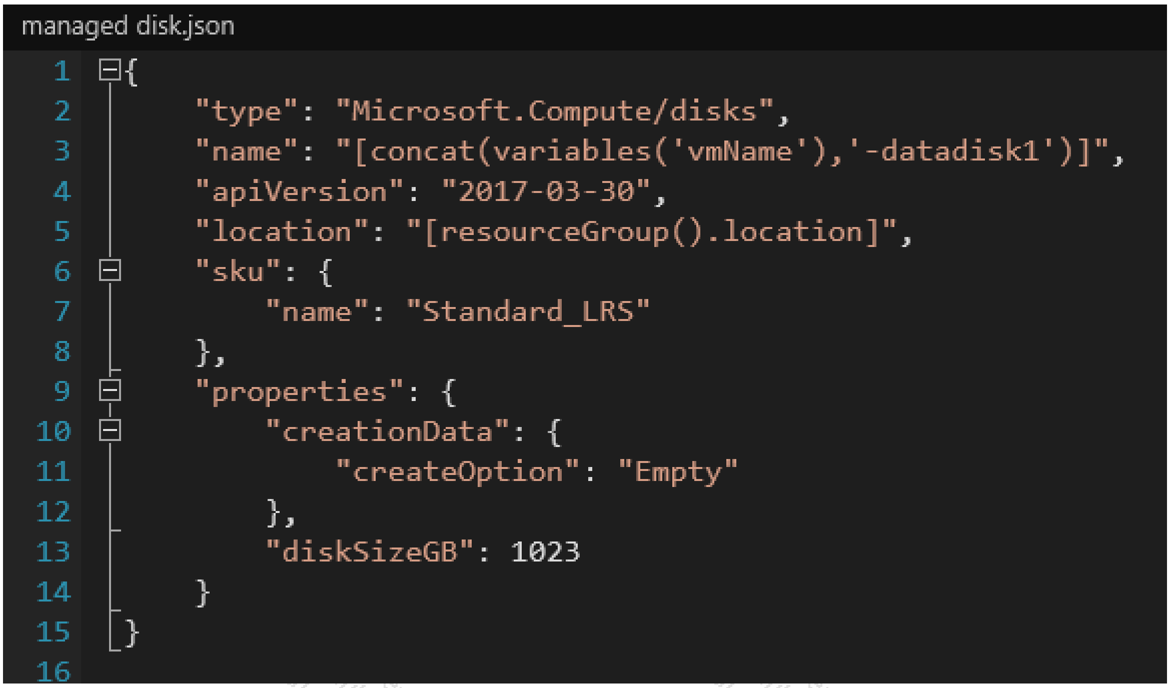
- Can deploy with templates
- Can manage with
  - PowerShell
  - Azure CLI
  - Portal
- Easy snapshot creation and management
- Rapid performance changes.

Deployment of Azure VM disks is easy to achieve using any of the standard tools.

- Azure Portal
- PowerShell
- Azure CLI

JSON templates may be used to create and format managed disks as well as un-managed disks

The data disks can be added within the VM creation template or defined stand alone as a top-level disk resource; this would then be attached when the VM is created.



```
{
    "type": "Microsoft.Compute/disks",
    "name": "[concat(variables('vmName'),'-datadisk1')]",
    "apiVersion": "2017-03-30",
    "location": "[resourceGroup().location]",
    "sku": {
        "name": "Standard_LRS"
    },
    "properties": {
        "creationData": {
            "createOption": "Empty"
        },
        "diskSizeGB": 1023
    }
}
```

FIGURE 6.4: MANAGED DISK IN ARM

Lesson 3: Files

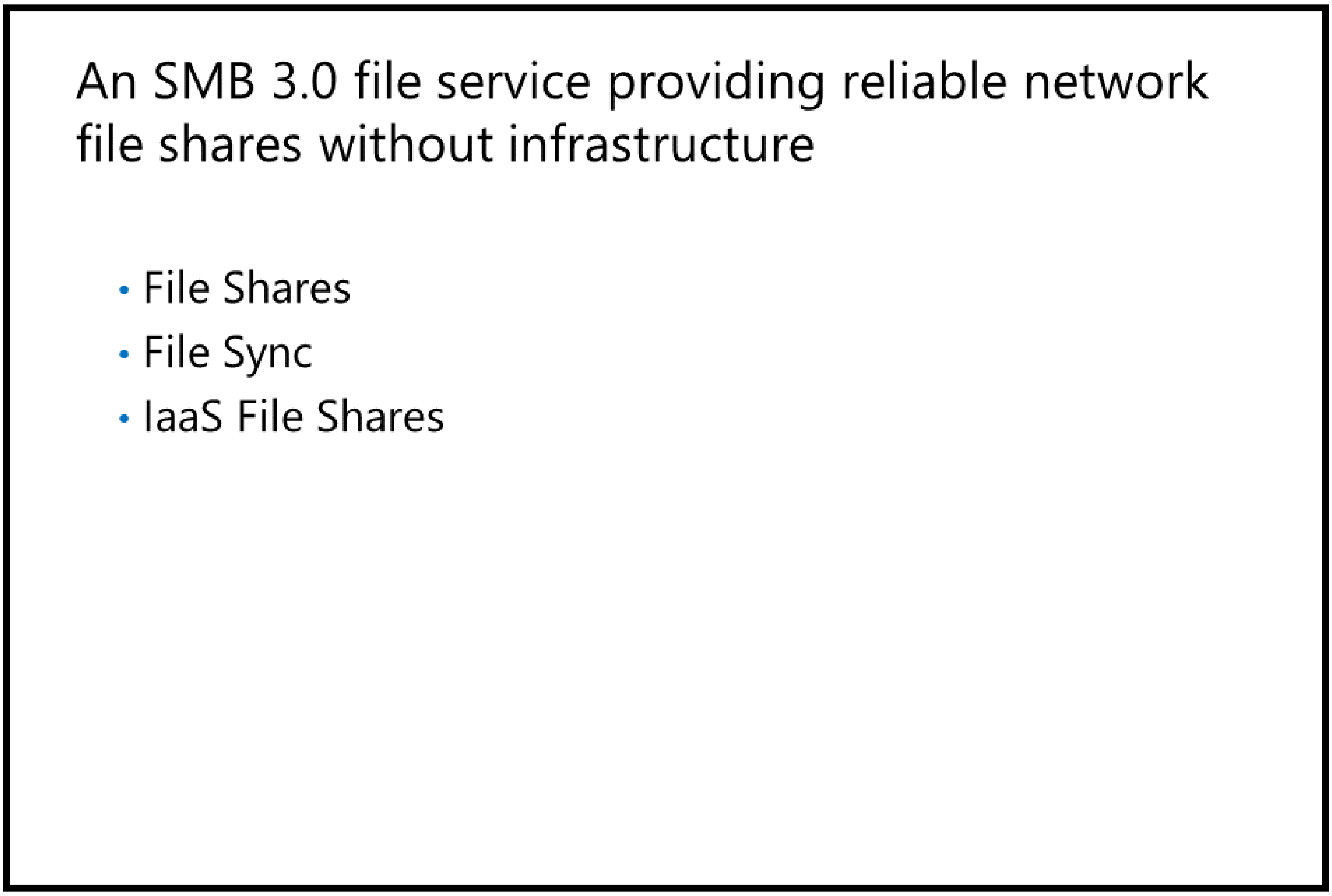
The ability to share files without the need to deploy the underlying server infrastructure provides several benefits when building an Azure based application. This lesson will describe the Azure File Service and the Azure File Sync service to allow the practitioner to choose which method of sharing files across an application or server infrastructure whether hybrid, on-premises or cloud-based.

Lesson objectives

After completing this lesson, you will be able to:

- Describe Azure Files service features.
- Decide which file sharing option suits and application best.
- Understand the benefits of Azure File Sync
- Create an Azure File Sync service.

Azure Files



An SMB 3.0 file service providing reliable network file shares without infrastructure

- File Shares
- File Sync
- IaaS File Shares

File storage offers shared storage for applications using the standard **SMB 3.0 protocol**. Microsoft Azure virtual machines and cloud services can share file data across application components via mounted shares, and on-premises applications can access file data in a share via the File storage API.

Applications running on Azure virtual machines or cloud services can mount a File storage share to access file data, just as a desktop application would mount a typical SMB share. Any number of Azure virtual machines or roles can mount and access the File storage share simultaneously.

Since a File storage share is a standard SMB 3.0 file share, applications running in Azure can access data in the share via file I/O APIs. Developers can, therefore, leverage their existing code and skills to migrate existing applications. IT Pros can use PowerShell cmdlets to create, mount, and manage File storage shares as part of the administration of Azure applications. This guide will show examples of both.

Typical uses of File storage include:

- Migrating on-premises applications that rely on file shares to run on Azure virtual machines or cloud services, without expensive rewrites
- Storing shared application settings, for example in configuration files
- Storing diagnostic data such as logs, metrics, and crash dumps in a shared location
- Storing tools and utilities needed for developing or administering Azure virtual machines or cloud services

File Share Service Components

File storage contains the following components:

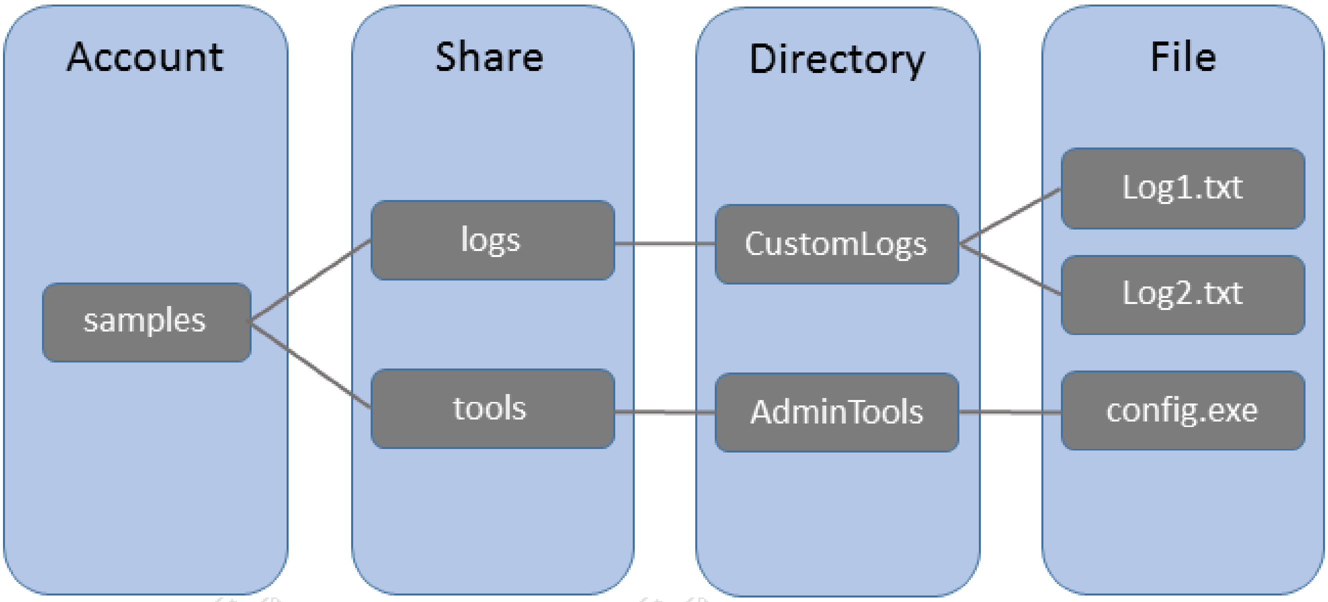
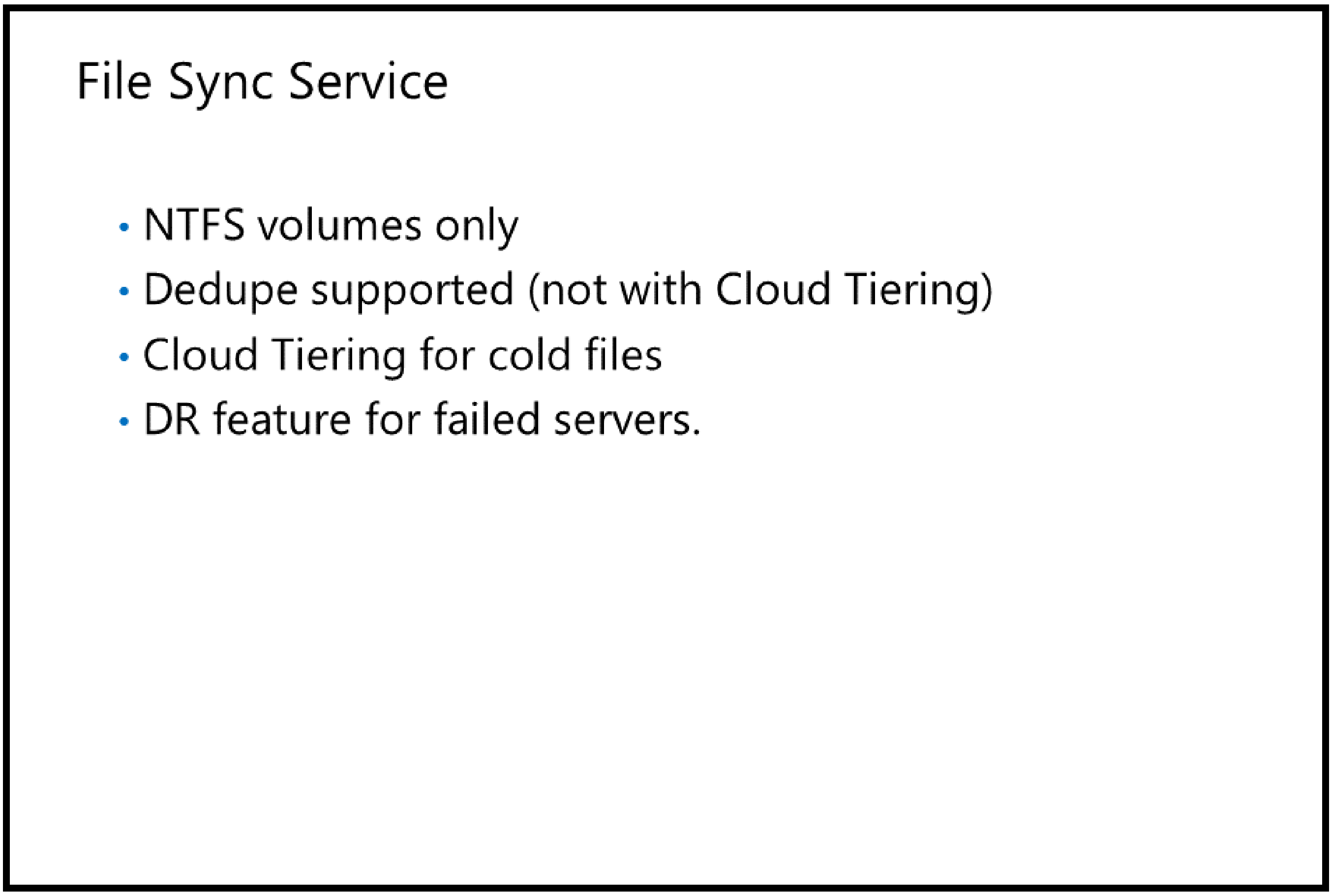


FIGURE 6.5: COMPONENTS OF FILE SHARE SERVICE

- **Storage Account**: All access to Azure Storage is done through a storage account.
- **Share**: A File storage share is an SMB 3.0 file share in Azure. All directories and files must be created in a parent share. An account can contain an unlimited number of shares, and a share can store an unlimited number of files, up to the capacity limits of the storage account.
- **Directory**: An optional hierarchy of directories.
- **File**: A file in the share. A file may be up to 1 TB in size.
- **URL format**: Files are addressable using the following URL format:https://[account].file.core.windows.net/<share>/<directory/directories>/<file>The following example URL could be used to address one of the files in the diagram above:HTTP://[account].file.core.windows.net/logs/CustomLogs/Log1.txt

Azure File Sync



File Sync Service

- NTFS volumes only
- Dedupe supported (not with Cloud Tiering)
- Cloud Tiering for cold files
- DR feature for failed servers.

The Azure File Sync Service is a new service that will allow you to centralize your file shares in Azure Files, whilst maintaining the compatibility of an on-premises file server with all the flexibility and performance benefits that provide. The Azure File Sync service turns your file server into a cache of the Azure-based file share. Any protocol installed on the Windows Server can access the file share, including SMB, NFS, and FTPS. The ability to have as many servers as you need spread across the globe turns the service into a useful way of providing all the benefits of a distributed file system without the infrastructure and maintenance requirements.

Azure File Sync terminology

When planning an Azure File Sync deployment, there are several components required to deploy the service efficiently.

- **Storage Sync Service**: The Storage Sync Service is the Azure resource created to host the Azure File Sync service. The Storage Sync Service resource is deployed to a resource group and can be created and amended using JSON templates if required. This service is required since the service can sync between multiple storage accounts, hence an additional resource is required to manage this. A subscription can contain multiple storage sync services.
- **Sync Group**: A Sync Group defines and controls the hierarchy and topology of the files to be synced. The sync group will contain Cloud and Server endpoints. Async service can contain multiple sync groups.
- **Registered Server**: Before adding a server endpoint to a sync group, the server must be registered with a storage sync service. A server can only be registered to a single sync service. Async service can host as many registered servers as you need.
- **Azure File Sync Agent**: To register a server, you need to install the Azure File Sync Agent.

This is a small downloadable MSI package comprising three components:

- **FileSyncSvc.exe**: Monitors changes on Server Endpoints, and for initiating sync sessions to Azure.
- **StorageSync.sys**: A file system filter, which handles tiering files to Azure Files.
- **PowerShell Management cmdLets**: PowerShell cmdlets for the **Microsoft.StorageSync** Azure resource provider.

- **Server Endpoint**: Once registered, you can add a server to a Sync group, this then becomes a server endpoint. A server endpoint is synonymous with a folder or a volume on the server that will cache the contents of the Azure File Share. Cloud tiering is configured individually by server endpoint.
- **Cloud Endpoint**: When added to a sync group, an Azure File Share is a cloud endpoint. One Azure File Share can only be a member of one Cloud Endpoint and thereby can only be a member of one Sync Group. Any files that exist in a cloud endpoint or a server endpoint before they are added to the sync group, automatically become merged with all other files in the sync group.

Here's a list of features available for Azure Files:

- **Adding files to the File Share**: Azure File Sync supports adding and removing files directly within the Azure file share. These files will only sync once every 24 hours down to the server endpoints. This is due to the change detection job only being scheduled once every 24 hours.
- **Cloud tiering**: Cloud tiering is a feature of Azure File Sync which can save considerable space on a server endpoint. When a file is tiered, the sync system filter replaces the local file with a pointer to the location if the Azure file share. This is marked as offline in NTFS. When accessed locally the file is downloaded and opened for use. This is Hierarchical Storage Management (HSM)
- **Supported versions of Windows Server**: Currently, Azure File Sync is supported by all GUI editions of Windows Server 2012 R2 and Windows Server 2016.ersions of Windows.
- **Access control lists (ACL)**: Supported and enforced on files held on Server endpoints. Azure Files do not currently support ACLs
- **NTFS compression**: Fully supported, and Sparse files are fully supported but are stored in the cloud as full files, and any cloud changes are synced as full files on server endpoints.
- **Failover Clustering**: Supported for File Server for General Use but not for Scale-out file server for application data. Cluster Shared Volumes are not supported. To function correctly, the sync agent must be installed on every node of a cluster.
- **Data Deduplication**: Fully supported for volumes that do not have cloud tiering enabled.

- **Encryption solutions**: Azure File Sync, is known to work with BitLocker Drive Encryption and Azure Rights Management Services. NTFS Encrypted File System does not work with Azure File Sync.

Lesson 4: StorSimple

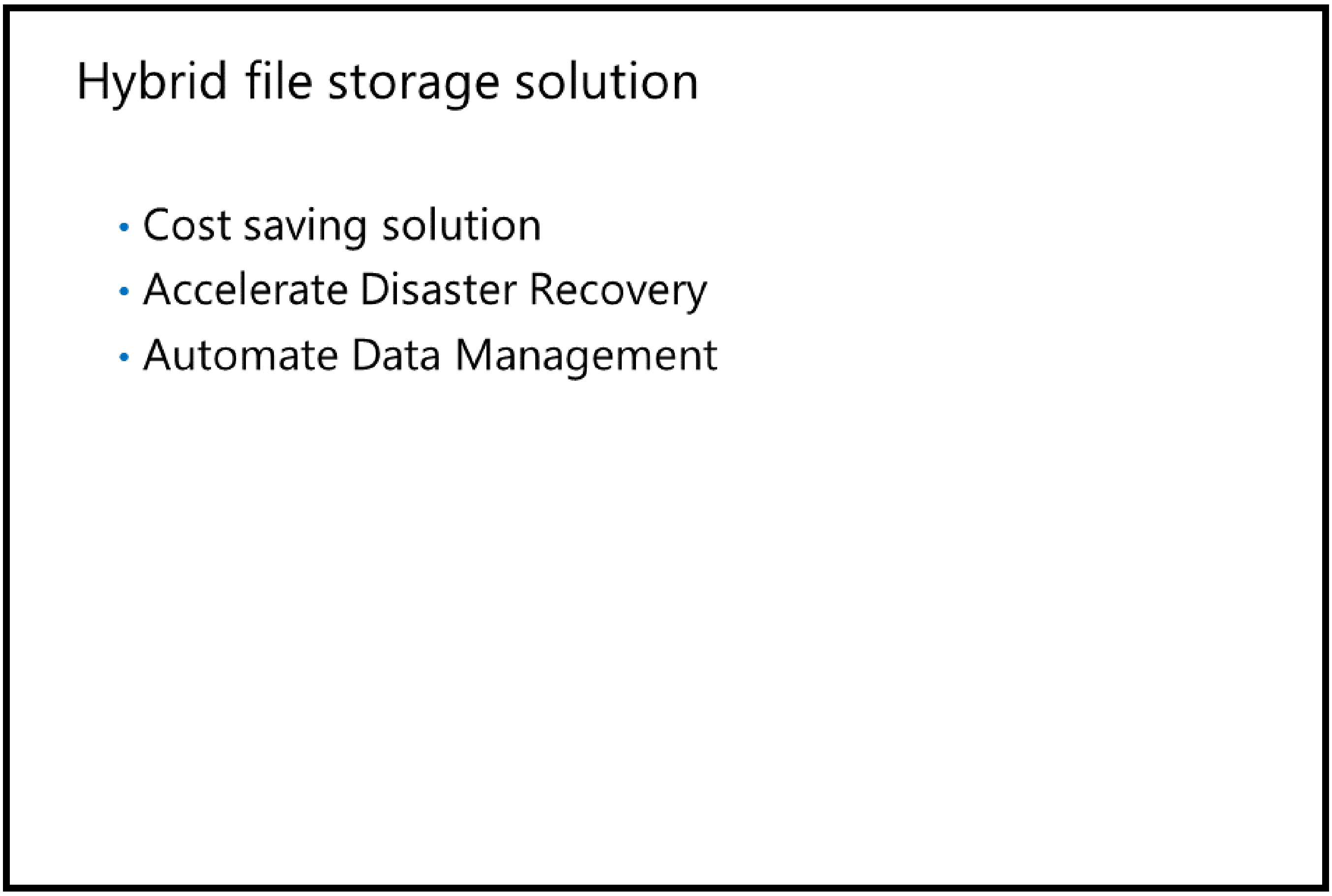
The use of Active Directory is widespread throughout the on-premises and cloud-based Windows infrastructure world. The advent of Azure AD brings many options for the Azure Architect to choose between. This lesson will examine the benefits of and differences between cloud only and hybrid solutions comprised of on-premises Active Directory Domain Services, Azure AD, and Azure AD Domain Services

Lesson objectives

After completing this lesson, you will be able to:

- Describe the StorSimple service and devices.
- Decide when to use StorSimple in your hybrid File storage solution.
- Understand the StorSimple data tiering process.

StorSimple



Hybrid file storage solution

- Cost saving solution
- Accelerate Disaster Recovery
- Automate Data Management

StorSimple is the combination of a service, device and management tools that can create workflows for migrating data to a cloud storage center or back on premise.

The StorSimple device is an on-premises hybrid storage array that provides primary storage and iSCSI access to data stored on it. It manages communication with cloud storage and helps to ensure the security and confidentiality of all data that is stored on the StorSimple solution. The StorSimple device includes solid state drives (SSDs) and hard disk drives (HDDs), as well as support for clustering and automatic failover. It contains a shared processor, shared storage, and two mirrored controllers.

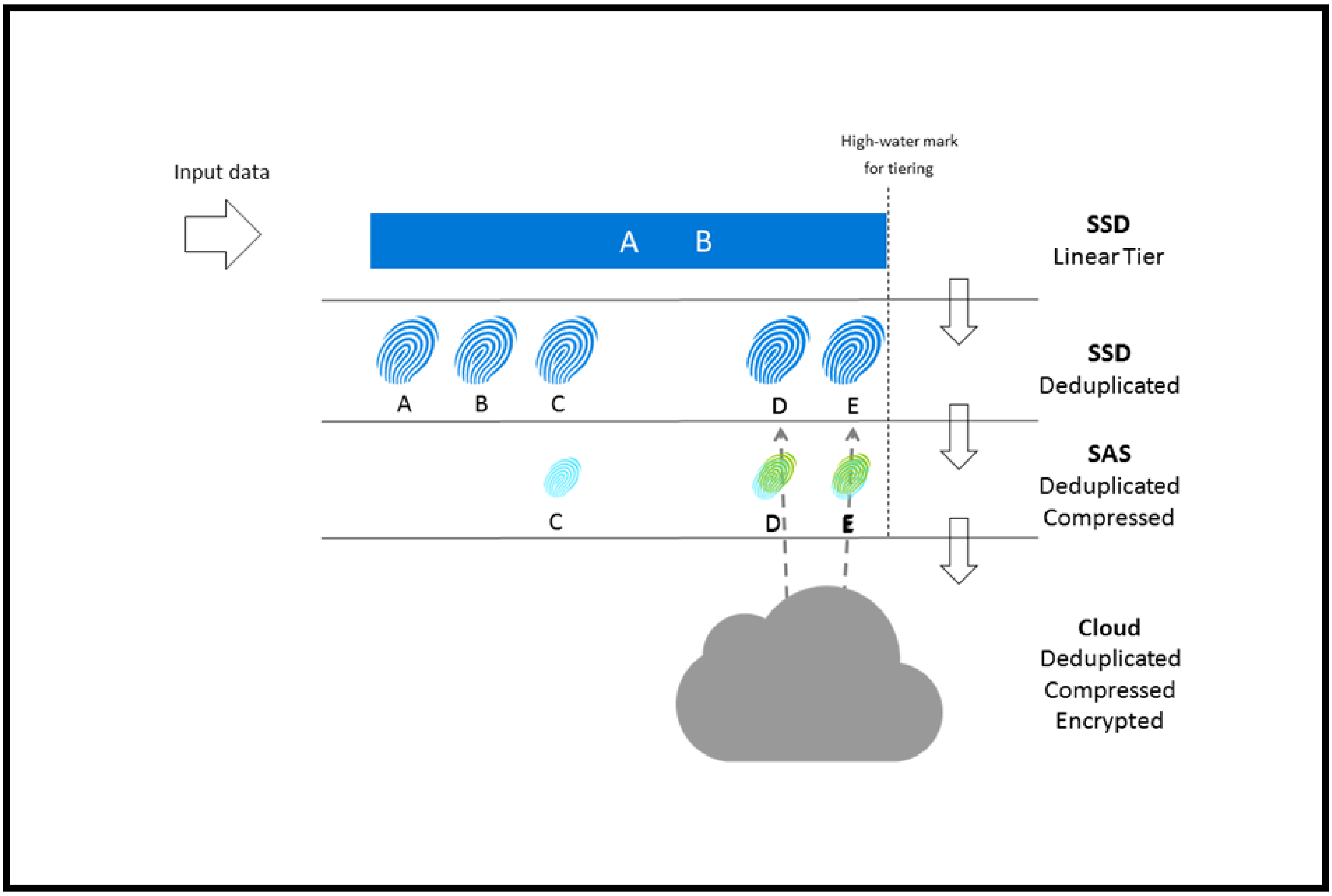
You can alternatively use StorSimple to create a virtual device that replicates the architecture and capabilities of the actual hybrid storage device. The StorSimple virtual device (also known as the StorSimple Virtual Appliance) runs on a single node in an Azure virtual machine.

StorSimple provides a web-based user interface (the StorSimple Manager service) that enables you to manage data center and cloud storage centrally. You can also use a Windows PowerShell-based, a command-line interface that includes dedicated cmdlets for managing your StorSimple device. Finally, you can interact with StorSimple using a Microsoft Management Console (MMC) snap-in that's used to configure and create consistent, point-in-time backup copies of local and cloud data.

Features

- **Transparent integration** – Microsoft Azure StorSimple uses the Internet Small Computer System Interface (iSCSI) protocol to invisibly link data storage facilities. This ensures that data stored in the cloud, in the data center, or on remote servers appears to be stored at a single location.
- **Reduced storage costs** – Microsoft Azure StorSimple allocates sufficient local or cloud storage to meet current demands and extends cloud storage only when necessary. It further reduces storage requirements and expense by eliminating redundant versions of the same data (deduplication) and by using compression.
- **Simplified storage management** – Microsoft Azure StorSimple provides system administration tools that you can use to configure and manage data stored on-premises, on a remote server, and in the cloud. Additionally, you can manage backup and restore functions from a Microsoft Management Console (MMC) snap-in. StorSimple provides a separate, optional interface that you can use to extend StorSimple management and data protection services to content stored on SharePoint servers.
- **Improved disaster recovery and compliance** – Microsoft Azure StorSimple does not require extended recovery time. Instead, it restores data as it is needed. This means regular operations can continue with minimal disruption. Additionally, you can configure policies to specify backup schedules and data retention.
- **Data mobility** – Data uploaded to Microsoft Azure cloud services can be accessed from other sites for recovery and migration purposes. Additionally, you can use StorSimple to configure StorSimple virtual devices on virtual machines (VMs) running in Microsoft Azure. The VMs can then use virtual devices to access stored data for test or recovery purposes.

Data Tiering



StorSimple automatically tiers and classifies your data based on how often you access it. Data is always being shuffled between tiers as the mechanism learns about your usage patterns.

Data that is most active is stored locally, while less active and inactive data is automatically migrated to the cloud. To enable quick access, StorSimple stores very active data (hot data) on SSDs in the StorSimple device. It stores data that is occasionally used (warm data) on HDDs in the device or on servers at the datacenter. It moves inactive data, backup data, and data retained for archival or compliance purposes to the cloud. StorSimple adjusts and rearranges data and storage assignments as usage patterns change. For example, some information might become less active over time. As it becomes progressively less active, it is migrated from SSD to HDD and then to the cloud. If that same data becomes active again, it is migrated back to the storage device.

Lab: Deploying Azure Storage to Support Other Workloads in Azure

Scenario

An established business has hired your company to help it transition from using Storage Accounts and blobs to managed disks in Azure.

Objectives

- Create an ARM Template that deploys a VM backed by a Storage Account
- Create an ARM Template that deploys a VM backed by a Managed Disk

Lab setup

Estimated Time: 60 minutes

Virtual machine: **20535A-SEA-ARCH**

User name: **Admin**

Password: **Pa55w.rd**

The lab steps for this course change frequently due to updates to Microsoft Azure. Microsoft Learning updates the lab steps frequently, so they are not available in this manual. Your instructor will provide you with the lab documentation.

Exercise 1: Create Required Resources for a Virtual Machine

Exercise 2: Create a VM With a Storage Account

Exercise 3: Create a VM With a Managed Disk

Exercise 4: Cleanup Subscription

Review Question(s)

Module review and takeaways

Review Question(s)