**Locally-Redundant Storage (LRS)**

LRS replicates data three times within one data center located in a primary region. When LRS is enabled, Azure Storage only registers write requests as successful once data is written to three replicas. LRS provides at least 99.999999999% durability for objects during a given year.

ZRS performs replication across three Azure availability zones. Each Azure availability zone is an individual physical location with its own independent networking, power, and cooling. ZRS provides a minimum of 99.9999999999% durability for objects during a given year.

**Geo-Redundant Storage (GRS)**

GRS provides additional redundancy for data storage compared to LRS or ZRS. In addition to the three copies of data stored in one region, there are three copies stored in a paired Azure region. So GRS provides all the features of LRS storage in the primary zone, and additionally, provides a secondary LRS data storage in another region.

There are two disadvantages of GRS redundancy:

* Replication between regions is asynchronous and so data is propagated with a small delay
* The second region cannot be accessed or read until the storage account fails over

**Read-Access Geo-Redundant (RA-GRS)**

RA-GRS has all the same level of redundancy of standard GRS replication, with an additional benefit—the secondary copies stored in paired Azure regions are readable. This means that if your application is configured correctly, you can use multiple readable endpoints. This increases the SLA for read operations to 99.99%.

However, the SLA for write operations remains 99.9%, because a single area still controls write and update operations.

Due to their asynchronous replication, both types of GRS replication have some replication delay. You can use the LastSyncTime parameter to ensure you are reading the latest copy of the data.

**Object Replication for Block Blob Storage**

The preceding replication methods were relevant for all Azure storage services. This is a special replication method which is available only for Block Blob Storage.

The object replication method is asynchronous. You can use it to automatically move data to an archive tier, in order to optimize data distribution and reduce costs. Or, you can use it to synchronize data to a storage resource running nearer to your users, to reduce latency.

Block blobs are replicated according to your replication policy, which specifies source/target Azure accounts and containers, and which block blobs should be replicated.

Block blob object replication copies:

* Blob content
* Blob version
* Blob metadata

**Geo-zone-redundant storage**

Geo-zone-redundant storage (GZRS) combines the high availability provided by redundancy across availability zones with protection from regional outages provided by geo-replication. Data in a GZRS storage account is copied across three [Azure availability zones](https://docs.microsoft.com/en-us/azure/availability-zones/az-overview) in the primary region and is also replicated to a secondary geographic region for protection from regional disasters. Microsoft recommends using GZRS for applications requiring maximum consistency, durability, and availability, excellent performance, and resilience for disaster recovery.

With a GZRS storage account, you can continue to read and write data if an availability zone becomes unavailable or is unrecoverable. Additionally, your data is also durable in the case of a complete regional outage or a disaster in which the primary region isn't recoverable. GZRS is designed to provide at least 99.99999999999999% (16 9's) durability of objects over a given year.

The following diagram shows how your data is replicated with GZRS or RA-GZRS:

Azure Storage is Microsoft's cloud storage solution for modern data storage scenarios. Azure Storage offers a massively scalable object store for data objects, a file system service for the cloud, a messaging store for reliable messaging, and a NoSQL store. Azure Storage is:

* Durable and highly available. Redundancy ensures that your data is safe during transient hardware failures. You replicate data across datacenters or geographical regions for protection from local catastrophe or natural disaster. Data replicated remains highly available during an unexpected outage.
* Secure. All data written to Azure Storage is encrypted by the service. Azure Storage provides you with fine-grained control over who has access to your data.
* Scalable. Azure Storage is designed to be massively scalable to meet the data storage and performance needs of today's applications.
* Managed. Microsoft Azure handles hardware maintenance, updates, and critical issues for you.
* Accessible. Data in Azure Storage is accessible from anywhere in the world over HTTP or HTTPS. Microsoft provides SDKs for Azure Storage in a various languages .NET, Java, Node.js, Python, PHP, Ruby, Go, and REST API. Azure Storage supports scripting in Azure PowerShell or Azure CLI. And the Azure portal and Azure Storage Explorer offer easy visual solutions for working with your data.

Azure Storage is a service that you can use to store files, messages, tables, and other types of information. You use Azure storage for applications like file shares. Developers use Azure storage for working data. Working data includes websites, mobile apps, and desktop applications. Azure storage is also used by IaaS virtual machines, and PaaS cloud services. You can generally think of Azure storage in three categories.

* Storage for Virtual Machines. Virtual machine storage includes disks and files. Disks are persistent block storage for Azure IaaS virtual machines. Files are fully managed file shares in the cloud.
* Unstructured Data. Unstructured data includes Blobs and Data Lake Store. Blobs are highly scalable, REST-based cloud object store. Data Lake Store is Hadoop Distributed File System (HDFS) as a service.
* Structured Data. Structured data includes Tables, Cosmos DB, and Azure SQL DB. Tables are a key/value, autoscaling NoSQL store. Cosmos DB is a globally distributed database service. Azure SQL DB is a fully managed database-as-a-service built on SQL.

General purpose storage accounts have two tiers: Standard and Premium.

* Standard storage accounts are backed by magnetic drives (HDD) and provide the lowest cost per GB. Use Standard storage for applications that require bulk storage or where data is infrequently accessed.
* Premium storage accounts are backed by solid-state drives (SSD) and offer consistent low-latency performance. Use Premium storage for Azure virtual machine disks with I/O-intensive applications, like databases.

*From <*[*https://docs.microsoft.com/en-us/learn/modules/configure-storage-accounts/2-implement-azure-storage*](https://docs.microsoft.com/en-us/learn/modules/configure-storage-accounts/2-implement-azure-storage)*>*

Azure Storage includes these data services, each of which is accessed through a storage account.

* Azure Containers (Blobs): A massively scalable object store for text and binary data.
* Azure Files: Managed file shares for cloud or on-premises deployments.
* Azure Queues: A messaging store for reliable messaging between application components.
* Azure Tables: A NoSQL store for schemaless storage of structured data.

Container (blob) storage

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data, such as text or binary data. Blob storage is ideal for:

* Serving images or documents directly to a browser.
* Storing files for distributed access.
* Streaming video and audio.
* Storing data for backup and restore, disaster recovery, and archiving.
* Storing data for analysis by an on-premises or Azure-hosted service.

Objects in Blob storage can be accessed from anywhere in the world via HTTP or HTTPS. Users or client applications can access blobs via URLs, the Azure Storage REST API, Azure PowerShell, Azure CLI, or an Azure Storage client library. The storage client libraries are available for multiple languages, including .NET, Java, Node.js, Python, PHP, and Ruby.

Azure files

Azure Files enables you to set up highly available network file shares that can be accessed by using the standard Server Message Block (SMB) protocol. That means that multiple VMs can share the same files with both read and write access. You can also read the files using the REST interface or the storage client libraries.

One thing that distinguishes Azure Files from files on a corporate file share is that you can access the files from anywhere in the world using a URL that points to the file and includes a shared access signature (SAS) token. You can generate SAS tokens; they allow specific access to a private asset for a specific amount of time.

File shares can be used for many common scenarios:

* Many on-premises applications use file shares. This feature makes it easier to migrate those applications that share data to Azure. If you mount the file share to the same drive letter that the on-premises application uses, the part of your application that accesses the file share should work with minimal, if any, changes.
* Configuration files can be stored on a file share and accessed from multiple VMs. Tools and utilities used by multiple developers in a group can be stored on a file share, ensuring that everybody can find them, and that they use the same version.
* Diagnostic logs, metrics, and crash dumps are just three examples of data that can be written to a file share and processed or analyzed later.

The storage account credentials are used to provide authentication for access to the file share. This means anybody with the share mounted will have full read/write access to the share.

Queue storage

The Azure Queue service is used to store and retrieve messages. Queue messages can be up to 64 KB in size, and a queue can contain millions of messages. Queues are used to store lists of messages to be processed asynchronously.

For example, if you want your customers to be able to upload pictures, and you want to create thumbnails for each picture. You could have your customer wait for you to create the thumbnails while uploading the pictures. An alternative would be to use a queue. When the customer finishes the upload, write a message to the queue. Then have an Azure Function retrieve the message from the queue and create the thumbnails. Each of the processing parts can be scaled separately, giving you more control when tuning it for your usage.

Table storage

Azure Table storage is now part of Azure Cosmos DB. In addition to the existing Azure Table storage service, there is a new Azure Cosmos DB Table API offering that provides throughput-optimized tables, global distribution, and automatic secondary indexes. Table storage is ideal for storing structured, non-relational data.

*From <*[*https://docs.microsoft.com/en-us/learn/modules/configure-storage-accounts/3-explore-azure-storage-services*](https://docs.microsoft.com/en-us/learn/modules/configure-storage-accounts/3-explore-azure-storage-services)*>*

Determine replication strategies

The data in your Azure storage account is always replicated to ensure durability and high availability. Azure Storage replication copies your data so that it is protected from planned and unplanned events ranging from transient hardware failures, network or power outages, massive natural disasters, and so on. You can choose to replicate your data within the same data center, across zonal data centers within the same region, and even across regions. Replication ensures that your storage account meets the Service-Level Agreement (SLA) for Storage even in the face of failures.

Comparison of replication options

The following table provides a quick overview of the scope of durability and availability that each replication strategy will provide you for a given type of event (or event of similar impact).

Locally redundant storage

LRS is the lowest-cost replication option and offers the least durability compared to other options. If a datacenter-level disaster (for example, fire or flooding) occurs, all replicas may be lost or unrecoverable.

Despite its limitations, LRS may be appropriate in these scenarios:

* When your application stores data that can be easily reconstructed if data loss occurs.
* When your data is constantly changing, for example a live feed, and storing the data is really not required.
* When your application is restricted to replicating data only within a country due to data governance requirements.

Zone redundant storage

Zone Redundant Storage (ZRS) synchronously replicates your data across three (3) storage clusters in a single region. Each storage cluster is physically separated from the others and resides in its own availability zone. Each availability zone, and the ZRS cluster within it, is autonomous, with separate utilities and networking capabilities. Storing your data in a ZRS account ensures that you will be able to access and manage your data if a zone becomes unavailable. ZRS provides excellent performance and low latency.

Here are a few of more things to know about ZRS:

* ZRS is not yet available in all regions.
* Changing to ZRS from another data replication option requires the physical data movement from a single storage stamp to multiple stamps within a region.
* ZRS may not protect your data against a regional disaster where multiple zones are permanently affected. Instead, ZRS offers resiliency for your data.

Geo-redundant storage

Geo-redundant storage (GRS) replicates your data to a secondary region (hundreds of miles away from the primary location of the source data). GRS costs more than LRS, but GRS provides a higher level of durability for your data, even if there is a regional outage. GRS is designed to provide at least 99.99999999999999% (16 9's) durability. When your storage account has GRS enabled, then your data is durable even when there is a complete regional outage or a disaster where the primary region isn't recoverable.

For a storage account with GRS or RA-GRS enabled, all data is first replicated with locally redundant storage (LRS). An update is first committed to the primary location and replicated using LRS. The update is then replicated asynchronously to the secondary region using GRS. When data is written to the secondary location, it's also replicated within that location using LRS. Both the primary and secondary regions manage replicas across separate fault domains and upgrade domains within a storage scale unit. The storage scale unit is the basic replication unit within the datacenter. Replication at this level is provided by LRS. If you opt for GRS, you have two related options to choose from:

* GRS replicates your data to another data center in a secondary region, but that data is available to be read only if Microsoft initiates a failover from the primary to secondary region.
* Read-access geo-redundant storage (RA-GRS) is based on GRS. RA-GRS replicates your data to another data center in a secondary region, and also provides you with the option to read from the secondary region. With RA-GRS, you can read from the secondary regardless of whether Microsoft initiates a failover from the primary to the secondary.

Geo-zone redundant storage

Geo-zone-redundant storage (GZRS) combines the high availability of zone-redundant storage with protection from regional outages as provided by geo-redundant storage. Data in a GZRS storage account is replicated across three Azure availability zones in the primary region and also replicated to a secondary geographic region for protection from regional disasters. Each Azure region is paired with another region within the same geography, together making a regional pair.

With a GZRS storage account, you can continue to read and write data if an availability zone becomes unavailable or is unrecoverable. Additionally, your data is also durable when a complete regional outage or a disaster in which the primary region isn’t recoverable. GZRS is designed to provide at least 99.99999999999999% (16 9's) durability of objects over a given year. GZRS also offers the same scalability targets as LRS, ZRS, GRS, or RA-GRS. You can optionally enable read access to data in the secondary region with read-access geo-zone-redundant storage (RA-GZRS).

Microsoft recommends using GZRS for applications requiring consistency, durability, high availability, excellent performance, and resilience for disaster recovery. Enable RA-GZRS for read access to a secondary region when there is a regional disaster.

**Review storage security strategies**

Azure Storage provides a comprehensive set of security capabilities that together enable developers to build secure applications.

* Encryption. All data written to Azure Storage is automatically encrypted using Storage Service Encryption (SSE).
* Authentication. Azure Active Directory (Azure AD) and Role-Based Access Control (RBAC) are supported for Azure Storage for both resource management operations and data operations, as follows:
* You can assign RBAC roles scoped to the storage account to security principals and use Azure AD to authorize resource management operations such as key management.
* Azure AD integration is supported for data operations on the Blob and Queue services.
* Data in transit. Data can be secured in transit between an application and Azure by using Client-Side Encryption, HTTPS, or SMB 3.0.
* Disk encryption. OS and data disks used by Azure virtual machines can be encrypted using Azure Disk Encryption.
* Shared Access Signatures. Delegated access to the data objects in Azure Storage can be granted using Shared Access Signatures.

Authorization options

Every request made against a secured resource in the Blob, File, Queue, or Table service must be authorized. Authorization ensures that resources in your storage account are accessible only when you want them to be, and only to those users or applications to whom you grant access. Options for authorizing requests to Azure Storage include:

* Azure Active Directory (Azure AD). Azure AD is Microsoft's cloud-based identity and access management service. With Azure AD, you can assign fine-grained access to users, groups, or applications via role-based access control (RBAC).
* Shared Key. Shared Key authorization relies on your account access keys and other parameters to produce an encrypted signature string that is passed on the request in the Authorization header.
* Shared access signatures. Shared access signatures (SAS) delegate access to a particular resource in your account with specified permissions and over a specified time interval.
* Anonymous access to containers and blobs. You can optionally make blob resources public at the container or blob level. A public container or blob is accessible to any user for anonymous read access. Read requests to public containers and blobs do not require authorization.

*From <*[*https://docs.microsoft.com/en-us/learn/modules/configure-storage-security/2-review-strategies*](https://docs.microsoft.com/en-us/learn/modules/configure-storage-security/2-review-strategies)*>*

Create shared access signatures

A shared access signature (SAS) is a URI that grants restricted access rights to Azure Storage resources. You can provide a SAS to clients who shouldn't have access to your storage account key. By distributing a SAS URI to these clients, you grant them access to a resource for a specified period of time. SAS is a secure way to share your storage resources without compromising your account keys.

A SAS gives you granular control over the type of access you grant to clients who have the SAS, including:

* An account-level SAS can delegate access to multiple storage services. For example, blob, file, queue, and table.
* An interval over which the SAS is valid, including the start time and the expiry time.
* The permissions granted by the SAS. For example, a SAS for a blob might grant read and write permissions to that blob, but not delete permissions.

Optionally, you can also:

* Specify an IP address or range of IP addresses from which Azure Storage will accept the SAS. For example, you might specify a range of IP addresses belonging to your organization.
* The protocol over which Azure Storage will accept the SAS. You can use this optional parameter to restrict access to clients using HTTPS.

Identify file sync components

To gain the most from Azure File Sync, it's important to understand the terminology.

Storage Sync Service. The Storage Sync Service is the top-level Azure resource for Azure File Sync. The Storage Sync Service resource is a peer of the storage account resource, and can similarly be deployed to Azure resource groups. A distinct top-level resource from the storage account resource is required because the Storage Sync Service can create sync relationships with multiple storage accounts via multiple sync groups. A subscription can have multiple Storage Sync Service resources deployed.

Sync group. A sync group defines the sync topology for a set of files. Endpoints within a sync group are kept in sync with each other. If for example, you have two distinct sets of files that you want to manage with Azure File Sync, you would create two sync groups and add different endpoints to each sync group. A Storage Sync Service can host as many sync groups as you need.

Registered server. The registered server object represents a trust relationship between your server (or cluster) and the Storage Sync Service. You can register as many servers to a Storage Sync Service instance as you want. However, a server (or cluster) can be registered with only one Storage Sync Service at a time.

Azure File Sync agent. The Azure File Sync agent is a downloadable package that enables Windows Server to be synced with an Azure file share. The Azure File Sync agent has three main components:

* FileSyncSvc.exe: The background Windows service that is responsible for monitoring changes on server endpoints, and for initiating sync sessions to Azure.
* StorageSync.sys: The Azure File Sync file system filter, which is responsible for tiering files to Azure Files (when cloud tiering is enabled).
* PowerShell management cmdlets: PowerShell cmdlets that you use to interact with the Microsoft.StorageSync Azure resource provider. You can find these at the following (default) locations:
* C:\Program Files\Azure\StorageSyncAgent\StorageSync.Management.PowerShell.Cmdlets.dll
* C:\Program Files\Azure\StorageSyncAgent\StorageSync.Management.ServerCmdlets.dll

Server endpoint. A server endpoint represents a specific location on a registered server, such as a folder on a server volume. Multiple server endpoints can exist on the same volume if their namespaces do not overlap (for example, F:\sync1 and F:\sync2). You can configure cloud tiering policies individually for each server endpoint. You can create a server endpoint via a mountpoint. Note, mountpoints within the server endpoint are skipped. You can create a server endpoint on the system volume but, there are two limitations if you do so:

* Cloud tiering cannot be enabled.
* Rapid namespace restore (where the system quickly brings down the entire namespace and then starts to recall content) is not performed.

Cloud endpoint. A cloud endpoint is an Azure file share that is part of a sync group. The entire Azure file share syncs, and an Azure file share can be a member of only one cloud endpoint. Therefore, an Azure file share can be a member of only one sync group. If you add an Azure file share that has an existing set of files as a cloud endpoint to a sync group, the existing files are merged with any other files that are already on other endpoints in the sync group.

**Locally-Redundant Storage (LRS)**

LRS replicates data three times within one data center located in a primary region. When LRS is enabled, Azure Storage only registers write requests as successful once data is written to three replicas. LRS provides at least 99.999999999% durability for objects during a given year.

ZRS performs replication across three Azure availability zones. Each Azure availability zone is an individual physical location with its own independent networking, power, and cooling. ZRS provides a minimum of 99.9999999999% durability for objects during a given year.

**Geo-Redundant Storage (GRS)**

GRS provides additional redundancy for data storage compared to LRS or ZRS. In addition to the three copies of data stored in one region, there are three copies stored in a paired Azure region. So GRS provides all the features of LRS storage in the primary zone, and additionally, provides a secondary LRS data storage in another region.

There are two disadvantages of GRS redundancy:

* Replication between regions is asynchronous and so data is propagated with a small delay
* The second region cannot be accessed or read until the storage account fails over

**Read-Access Geo-Redundant (RA-GRS)**

RA-GRS has all the same level of redundancy of standard GRS replication, with an additional benefit—the secondary copies stored in paired Azure regions are readable. This means that if your application is configured correctly, you can use multiple readable endpoints. This increases the SLA for read operations to 99.99%.

However, the SLA for write operations remains 99.9%, because a single area still controls write and update operations.

Due to their asynchronous replication, both types of GRS replication have some replication delay. You can use the LastSyncTime parameter to ensure you are reading the latest copy of the data.

**Object Replication for Block Blob Storage**

The preceding replication methods were relevant for all Azure storage services. This is a special replication method which is available only for Block Blob Storage.

The object replication method is asynchronous. You can use it to automatically move data to an archive tier, in order to optimize data distribution and reduce costs. Or, you can use it to synchronize data to a storage resource running nearer to your users, to reduce latency.

Block blobs are replicated according to your replication policy, which specifies source/target Azure accounts and containers, and which block blobs should be replicated.

Block blob object replication copies:

* Blob content
* Blob version
* Blob metadata

**Geo-zone-redundant storage**

Geo-zone-redundant storage (GZRS) combines the high availability provided by redundancy across availability zones with protection from regional outages provided by geo-replication. Data in a GZRS storage account is copied across three [Azure availability zones](https://docs.microsoft.com/en-us/azure/availability-zones/az-overview) in the primary region and is also replicated to a secondary geographic region for protection from regional disasters. Microsoft recommends using GZRS for applications requiring maximum consistency, durability, and availability, excellent performance, and resilience for disaster recovery.

With a GZRS storage account, you can continue to read and write data if an availability zone becomes unavailable or is unrecoverable. Additionally, your data is also durable in the case of a complete regional outage or a disaster in which the primary region isn't recoverable. GZRS is designed to provide at least 99.99999999999999% (16 9's) durability of objects over a given year.

The following diagram shows how your data is replicated with GZRS or RA-GZRS:

**File Shares**

Cloud based SMB or NFS file share. Accessible from W, Linux, Mac OS

Clients use 445. Need to know requirements: performance, sizing, redundancy. GPV2 and FileStorage are best options generally. Main difference is performance and FileStorage not having Geo-redundancy

Azure file sync – sync files between azure and on-prem servers

Sync group -