# Manage Azure identities and governance

**Managing Azure AD objects allows for tasks such as:**

* Creating users and groups
* Managing licenses in Azure AD
* Setting up administrative units
* Modifying user and group properties
* Configuring device settings and identity
* Performing bulk updates
* Managing guest accounts
* Enabling self-service password reset (SSPR)

**Azure Active Directory (AD)**Azure AD is a cloud-based directory and identity and access management (IAM) service, supporting user access to:

* Internal resources and apps on the corporate network
* Cloud apps developed by the organization
* External resources (e.g., Microsoft 365, Azure portal, and other SaaS apps)

**Azure AD Key Terms**

* **Identity**: An object that can be authenticated, such as a user with a username and password, or applications/services requiring certificate-based authentication.
* **Account**: An identity associated with data.
* **Azure AD Account**: Identity created via Azure AD or Microsoft 365, often referred to as a work or school account.
* **Azure Tenant**: A unique, trusted instance of Azure AD that represents an organization.
* **Azure Subscription**: A unit tied to a tenant, used for billing Azure services.
* **Cloud Identity**: A user account defined only in Azure AD.
* **Directory-Synchronized Identity**: Accounts synchronized from on-premises Active Directory using Azure AD Connect.
* **Guest User**: External accounts, such as from other cloud providers or Microsoft accounts, used to grant access to Azure resources.

### **Azure AD Terms**

Identity: an object that can be authenticated.

* A user with a username & password.
* Applications or other services that require authentication using certificates or private keys.

Account: an identity that has data associated with it.

* A valid identity is required to have an account.

Azure AD account: an identity created through Azure AD or M365.

* Identities are stored in Azure AD and are accessible to the org's cloud service subscriptions.
  + aka a work or school account

Azure tenant: a single, dedicated, and trusted instance of Azure AD.

* Each tenant, or directory, represents one organization.
  + When an org signs up for a Microsoft cloud service subscription, a new tenant is created automatically.
* Multiple tenants can be created.

Azure subscription: used to pay for Azure services.

* Each subscription is connected to one tenant, but orgs can have multiple subscriptions.

Cloud identity: a user account with a cloud identity is defined only in Azure AD.

* These user accounts includes admin accounts & users who are managed.
* A cloud identity can be for user accounts defined in the tenant and for user accounts defined in external Azure AD instances.
* When a cloud identity is removed from the main directory, the user account is deleted.

Directory-synchronized identity: user accounts that have a directory-synchronized identity are defined in an on-prem AD.

* Synchronizations occur using Azure AD Connect to bring the on-prem user accounts into Azure AD.
* Windows Server AD is the source of these user accounts.

Guest user: user accounts defined outside of Azure.

* User accounts from other cloud providers, and Microsoft accounts (Xbox).
* The source for guest users is Invited user.
* Guest user accounts are helpful when working with external vendors and contractors that need to gain access to your Azure resources.
  + Note: it's important to regularly review your org's guest user accounts.

**Azure AD vs. Active Directory Domain Services (AD DS)**While Azure AD provides cloud-based identity management, AD DS offers traditional on-premises directory services. Azure AD focuses on HTTP/S-based apps, supports third-party federation services, and features a flat structure (no organizational units or group policy objects).

**Azure AD Service Plans**

* **Azure AD Free**: Supports up to 500,000 directory objects, includes SSO and basic IAM features.
* **Premium P1**: Adds features like dynamic groups, self-service group management, and hybrid identity support.
* **Premium P2**: Provides advanced identity protection, governance (PIM), and risk-based conditional access.

### **Azure AD Plans**

Azure AD Free

* Up to 500,000 directory objects
* SSO: unlimited
* Core identity and access management (IAM)
* Business-to-business (B2B) collaboration

M365 Apps

* Unlimited directory objects
* SSO: unlimited
* Core IAM
* B2B collaboration
* IAM for M365 apps

Premium P1

* Unlimited directory objects
* SSO: unlimited
* Core IAM
* B2B collaboration
* IAM for M365 apps
* Premium features: dynamic groups, self-service group management, and cloud write-back capabilities.
  + Additionally adds the feature for SSPR for on-prem users.
* Hybrid identities: Microsoft Identity Manager, which is an on-prem IAM suite.
* Advanced group access management
* Conditional access

Premium P2

* Unlimited directory objects
* SSO: unlimited
* Core IAM
* B2B collaboration
* IAM for M365 apps
* Premium features
* Hybrid identities
* Advanced group access management
* Conditional access: risk-based
* Identity protection
* Identity governance: Privileged Identity Management (PIM) to help discover, restrict, and monitor admins & their access.
  + Provides just-in-time (JIT) access when needed.

### **Azure AD SSPR**

Self-service password reset (SSPR) allows users to have the ability to skip contacting the helpdesk and reset their passwords on their own.

* To enable SSPR, a global admin account is required.
  + Global admins are able to reset their own passwords, regardless of this setting being enabled for all users, selected or none.
    - Note: these are the only three options available for SSPR.
* SSPR uses a security group to limit users who have the privileges, if only some are enabled.
* All user accounts must have a license to use SSPR.
* At least one authentication method is required to reset a password.
  + To have a stronger plan, you should offer multiple methods for users to reset their passwords with.
  + Options include:
    - SMS
    - Email
    - Security code sent to a mobile or office phone
    - Security questions
  + Security questions can be required to registered by users in the tenant.
  + The number of correctly answered security questions required for a successful password reset can be configured.

### **Manage licenses in Azure AD**

### Admins can view and manage license plans, assign licenses to users or groups, and monitor expiring licenses.

View license plans & details

* Total: the total number of licenses purchased
* Assigned: the number of assigned licenses to users
* Available: the number of licenses thare are available to assign, including any that are about to expire
* Expiring soon: the number of licenses expiring soon

Assign licenses to a user or group

* User location must be specified before the license is assigned to a user
* If licenses are assigned from the user's license page, it can be directly assigned to a user

Remove or modify a license plan for users or groups

* On the licensed user / licensed group page, you can select the user or group that no longer has a license and click remove license
  + Note: licenses that a user inherits from a group **cannot** be removed directly

## **Azure AD admin units**

* Add admin units from the Azure portal
* Add details about users or groups in the unit
* Add assignments with relevant permissions per the admin unit
* Add IT members to assignments (roles)

### **Manage device settings and device identity** Azure AD supports managing device settings for various device types (Azure AD Joined, Registered, and Hybrid Joined Devices).

Azure AD Registered Devices

Azure AD Joined Devices

Hybrid Azure AD Joined Devices

### **Perform bulk updates**

### Azure AD supports bulk updates using CSV files or scripts.

For bulk user creation, you can use the following:

* CSV
* Scripting

Requires Global admin or User admin.

* Note: from experience, this isn't always the case, but for the exam, remember that's what Microsoft states is "required" to complete bulk updates.

## **Manage access control**

* Create custom role-based access control (RBAC) and Azure AD roles
* Provide access to Azure resources by assigning roles at different scopes
* Interpret access assignments

### **RBAC Scope** RBAC in Azure allows custom role creation and resource access management at various scopes, such as management groups, subscriptions, and resource groups.

When granting access at the parent scope, the permissions are then inherited to the child scopes.

In Azure, we can specific scope at the following levels:

* Management Group (apply policies to multiple subscriptions within groups)
  + Subscription (manage billing)
    - Resource Group (lifecycle management tier for resources)
      * Resource

### **Built-in RBAC Roles / Assignments**

* Owner has full access to **all** resources and can delegate access to other users.
* Contributor can create and manage **all**types of Azure resources, but cannot grant access to other users.
* Reader is able to view Azure resources.
  + VM Reader can be used to target VMs specifically.
* User access administrator manages user access to Azure resources.
  + Reminder: this is an Azure AD role, unlike Owner, Contributor, and Reader.

Note: Custom roles can be created.

## **Manage Azure subscriptions and governance** Resource groups in Azure allow for logical resource management, while management groups provide hierarchical organization of subscriptions to apply policies, budgets, and access controls across projects.

* Configure and manage Azure Policy
* Configure resource locks
* Apply and manage tags on resources
* Manage resource groups
* Manage subscriptions
* Manage costs by using alerts, budgets, and recommendations
* Configure management groups

### **Manage subscriptions** Subscriptions in Azure are logical units for billing and resource management, and can be grouped or isolated by project, environment, or team needs.

An Azure subscription is a logical unit for setting up a resource boundary, environment boundary, and billing boundary. Each subscription has an account attached to it that can be a work or school account, or an account Azure AD trusts.

* You can also use a Microsoft account to use Azure since Azure AD trusts these accounts.

An Azure account is *any* identity that's part of Azure AD, or a directory trusted by Azure AD.

* Note: if you sign up for an Azure account using a work or school account, any subscription will be created in the Azure AD your work or school account is part of.
  + When using a Microsoft account (or a personal account), Azure automatically generates an Azure AD tenant during the creation of your Azure account.

When you create an Azure account, you become the Account Administrator. By default, the account admin will have access to the new subscription.

* In order to grant others access, you will need to use role based access control (RBAC).
* Note: Microsoft recommends using RBAC to grant access to users and any external users to Azure resources as classic roles are not being used anymore.

Multiple subscriptions can exist within the same Azure account. Some examples of reasons you'd want multiple subscriptions include, but are not limited to:

* Project isolation
* Environment isolation

There are several methods in which you can create an Azure subscription. Some of the following options may not be available for you as specific eligibility depends on terms & conditions of each offer:

* Enterprise Agreements (EAs)
* Web Direct
* Partners
* Resellers

To see other offers supported by Azure, see [Offer details](https://azure.microsoft.com/en-in/support/legal/offer-details/).

Important:

* Invoices can be viewed on a monthly basis

### **Cost management**

Cloud governance includes managing your cloud cost. There are tools that can help you breakdown and track your costs. Azure Cost Management tools help monitor, forecast, and optimize cloud spending, supporting resource tagging, budget alerts, and cost trend analysis, including but not limited to:

* Complete billing administrative tasks
* Create budgets and set alerts upon specific thresholds being passed
* Predict future costs using analytics
* Export data to Power BI dashboards using the Azure Cost Management connector
  + Note: this is only available to EA/MCA customers.
* Calculate potential savings and track reservation usage
* Schedule usage reports to be exported to a storage account for auditing
* Ingest AWS costs & analyze them within Azure

Azure Cost Management can be accessed from the Azure portal and will display charts that help explain your cloud spending.

Remember:

* Costs are **resource-specific**
* Usage costs may vary between locations
* Costs for inbound & outbound data transfer differ
* We can pre-pay using **Azure reserved instances** to reduce our costs.
* We can use on-prem licenses with Azure Hybrid Benefit.
* Using tags, we can view spending trends to ensure we don't exceed budgets.
* Azure Advisor can be used to view cost optimization recommendations, helping us reduce trends.
* We can scale activities to help reduce the spend for our budgets.

### **Manage resource groups**

A resource group is a container used for the logical organization of Azure resources. These make deploying, deleting, and/or updating resources in bulk an easy task, saving you time from having to manually apply any changes to the resources.

* Note: it's advised to use caution before deleting resource groups as you are unable to undo this action.

Resources can be part of the same solution or grouped together however you prefer. Resources groups can also include resources from different regions, but this means the metadata will be stored in the same region as the resource group is located in, as resource groups contain the metadata about any resources making up the group.

* Note: grouping resources together for the same solution, or by the same type of resources, can help make managing them easier.

Important:

* Resource groups **cannot** be renamed.
* Resource groups **cannot** be nested.

### **Management groups**

Management groups provide the ability to logically group Azure subscriptions together, offering a new scope above the subscriptions to be used to:

* Grant access
* Assign policies
* Analyze costs

Cost management can be scoped at this level for tracking the costs of multiple subscriptions under one view. This includes creating budgets, which is particularily helpful for teams working on multiple projects under different subscriptions.

Remember:

* These groups provide organizational alignment for Azure subscriptions through *custom hierarchies* and *grouping*.
* We are enabled to target policies & budgets across subscriptions and inheritance down the hierarchies.
* By using optional management groups, we can support compliance & cost reporting.

With your tenant, there will be a **default**management group that is referred to as the *root management group*. Creating new management groups can be performed from the Azure portal, PowerShell, or Azure CLI, and these new groups will be created as *children* of the default management group.

* Note: the root management group **cannot**be changed, but it is *optional* to use.

Members of each management group will be the subscription, or multiple subscriptions which can be **nested** between management groups.

There are two required parameters when you create a new management group:

* Management Group ID: used to specify the management group when running commands against it.
  + Note: this cannot be changed after creating the management group.
* Display name: used to point to the management group when using PowerShell, Azure CLI, or REST API calls.

### **Azure policy**

This service allows you to create, assign, and manage policies with different rules, over different resources and scopes. Rules can be enforced to ensure the org meets any compliance controls necessary.

Azure policy will run evaluations constantly to resources are compliant and can make sure new resources are stopped before breaking any compliance requirements.

* Note: Azure policy cannot delete resources that are reported as noncompliant, however, we can resolve any that are noncompliant.

Azure policy features include:

* Applying policies at scale
  + Policies can be applied to the management group level to be inherited by all subscriptions that are part of the group.
* Compliance and enforcement
  + Built-in or custom policies can be used to ensure compliance. Resources that don't meet compliance with your policies won't be deployed.
* Mitigation and remediation:
  + Remediation can be automated or performed manually.

Azure policy can help with the following scenarios:

* Prevent users from deploying expensive services (Cosmos DB, ExpressRoute, etc.)
* Require specific tags during resource deployment
* Require Azure Backup service is enabled on *all* VMs
* Limit VM deployment to only contain a specific set of SKUs, which helps prevent expensive VM SKUs from being deployed.
* Require resources to be deployed in specific regions, which helps the org meet any data residency requirements.

Remember, Azure policy helps with the following:

* Managing policy definitions and initiatives at different levels of scope.
* Enforcing tags to ensure the policy is followed upon resource creation.
* Requiring resources to be provisioned with a naming convention that is predetermined.
* Restricting resource creation to a list of specified types.

To configure Azure Policies:

* Search for Policy from the Azure portal
* Select the Definitions pane from the Authoring section
* Click + Policy definition
* To set the Definition location, click the blue ellipsis box
* Provide a name, description, & category
* For Policy rules, create the policy using JSON
* Assign the policy to the resource using the assign option on the police service panel

Note: we can group various policies in a container.

* We can save existing policies to create new policy definitions that are assigned to new resources, using a new name.
  + The JSON can be used as a template to create a custom policy.
  + Tag Names will be a parameter in the policy, which require a display name before the policy can be created.

### **Azure policy definition**

To define policies: Azure portal > Policy > Definitions

* Note: add a filter to view built-in policies.
  + Using built-in policies will help reduce the time needed to write new policies.

The definition describes the action the policy will take when assigned. It's importation to note that this is in JSON format.

The following information is required while defining the policy:

* Definition location
  + This is either the subscription or management group.
  + Note: policies that are defined at the subscription level cannot be assigned to management groups.
* Name
  + This will be the name that displays in the policy list.
* Description
  + These will help others understand what the policy is for.
* Category
  + These are helpful to filter policies.
* Policy rule
  + We have two options: Import from GitHub or write in the Azure portal using the editor.

Policies will contain a condition and effect.

* Condition: what is evaluated to confirm if the effect is applied or not.
* Effect: if the condition is matched, this is the action that is taken.

### **Azure policy assignment & scope**

Policy assignment is the process of assigning a definition to a specific scope. Once the assign page is selected, we can select the scope for the policy from the available scopes listed.

* Note: it takes ~30mins for the evaluation report to be generated after assignment.
  + To assign custom policies to the management group, use the management group scope.

While assigning policies, the following information can be added on the Basics tab:

* Scope
  + This is where you would like the policy to apply.
* Exclusions
  + If any resources should be excluded, you would define those here.
  + Note: resources that are excluded will not be reported if they are noncompliant.
* Assignment name
  + This will be the name of the policy by default, however, it can be changed.
* Description
  + This will help others understand the reason for the assignment.
* Policy enforcement
  + Disabled or enabled.
* Assigned by
  + By default, this will be the user that's currently assigning the policy.

Policy scope is the process that determines which subscriptions, resource groups, and resources are enforced by the policy. Management groups, subscriptions, and resource groups are included in supported scopes.

### **Azure policy evaluation**

A dashboard is provided to view the overall compliance of selected scopes, allowing you to remediate noncompliance. To view this dashboard: Policy > Overview > Compliance

### **Resource locks**

There are two types of resource locks:

* Delete: Authorized users can read and modify a resource, but are unable to delete the resource.
* Read-only: Authorized users can read a resource but cannot delete or update the resource.

To delete a locked VM, users will be required to contact an Owner or a User Access Admin.

Note: if the resource lock is on at the resource group level, all resources within the resource group inherit the lock.

### **Apply & manage tags on resources**

Resource tags are a way to understand costs spanning across different departments and Azure scopes. These tags will provide metadata for the Azure resourcs and *name value pairs*.

Tags can be added in multiple ways:

* Azure portal
* CLI
* PowerShell
* ARM Templates
* REST API

Limitations:

* There is a limit of 50 tag name value pairs for each resource, resource group, or subscription.
* Names **cannot** include special characters such as <, >, %, &, \, /, or ?
* Tag names are limited to no more than 512 characters and values are limited to no more than 256 characters.
  + Important: storage account tag names are limited to no more than 124 characters.'

Note: tag values **can** be JSON strings, which can contain many values that apply to a single name.

* These would be managed for ARM templates.

# Implement and manage storage

## **Configure access to storage**

* Configure Azure Storage firewalls and virtual networks
* Create an use shared access signature (SAS) tokens
* Configure stored access policies
* Manage access keys
* Configure identity-based access for Azure Files

### **Azure Storage**

Azure Storage allows you to store files, messages, tables, and additional types of information in a durable, secure, scalable, managed, and accessible way.

Azure Storage allows us to store:

* Disks and files for VMs
  + HDDs (lowest cost) or SSDs (through Premium storage)
* Unstructured data
  + Blobs (binary large object files), these are highly scalable REST-based cloud object stores
  + Data lake storage, Hadoop for example, allow us to manage file systems as a service
* Structured data
  + Tables - key value, autoscaling NoSQL stores
  + Cosmos DB is a globally distributed database service
  + Azure SQL DB
  + Additional options for relational data
    - Important: structured data is nonrelational

Azure Storage Services: Azure Containers

* Allow us to store & manage scalable object store for unstructured text and binary data
  + Important: Blobs **cannot** be mounted.
  + We can create an unlimited number of blobs.

Azure Files

* Allow us to store & manage file shares for cloud **or** on-prem deployments
  + Directories

Azure Tables

* Allow us to use NoSQL store for schema-less storage of structured data
  + Entities

Azure Queues

* Allow us to manage reliable communications between application components
  + Messaging

**Important: every object stored in Azure Storage has it's own, unique URL.**

* The storage account name will form the *subdomain* of the URL
* The subdomain & domain forms **an endpoint**; examples of each endpoint:
  + Containers: http://storageacctname.blob.core.windows.net
  + Files: http://storageacctname.file.core.windows.net
  + Tables: http://storageacctname.table.core.windows.net
  + Queues: http://storageacctname.queue.core.windows.net

Note: custom domains can be configured - We can purchase a domain name from a registrar. - Once the domain name is purchased, we will need to validate it before adding it can be used with any services. - Verify the domain with either a TXT record, or MX record. - After verifying, we can map the CNAME record to our *target* to change the endpoint to display the custom domain instead of the storage account name.

### **Shared Access Signature (SAS) Tokens**

These tokens provide secure and delegated access to resources in our Azure Storage account.

* Provides access to clients without requiring sharing the storage account keys.
* Provides access to Azure resources in **multiple** storage accounts.
* Delegates secured access to a resource in **only one** of the storage services.

Best practices:

* Expiry date/time
* Time zone: to align the access required for users to their time zone
* Limit the distribution of SAS tokens
* Using HTTPS instead of HTTP
* Tokens should last only for the specific time needed to follow the principle of least privilege

### **Stored Access Policies**

Configuring stored access policies allows us to to have an additional level of control over *service level* SAS on the server side.

Stages of configuring stored access policies:

Create or modify stored access policy

* We can set a max of 5 access policies on a container, table, queue, or shared at any given time.
* The change, or update, of a stored access policy can take up to 30 seconds to apply before it's active.
  + Note: the SAS associated with the policy may display a 403 (Forbidden) error code.

Revoke or modify stored access policy

* To revoke, we can delete or rename the policy.
  + This can be done by modifying the signed identifier OR by changing the expiry date/time with a value in the past.

### **Storage Security**

The security capabilities of Azure Storage work together to allow developers to build secure apps.

Authorization options include:

* Azure AD: Cloud-based identity & access management option which gives control over resources.
* Shared Access Signatures: allow us to delete access to a specific resource for a predetermined amount of time with specific permissions.
* Shared keys: provides authorization & relies on access keys, along with additional params, to provide encrypted signature strings that are passed in request headers.
* Anonymous access to containers & blobs: only use when information is **not**considered sensitive (PII).

## **Configure and manage storage accounts**

* Create and configure storage accounts
* Configure Azure Storage redundancy
* Configure object replication
* Configure storage account encryption
* Manage data by using Azure Storage Explorer and AzCopy

### **Storage Redundancy**

Azure Storage **always** stores multiple copies of data to ensure it's protected from planned and unplanned events, such as:

* Network outages
* Power outages
* Transient hardware failures
* Natural disasters

Azure Storage Redundnacy provides the ability to choose the redundancy options that meet our availability and durability needs.

Important: the redundancy settings for a storage account is shared for **all** storage services exposed by the storage account.

* **All** storage resources deployed in the same storage account will have the **same**redundancy setting.

Redundancy in Primary region: data is replicated **three** times in the primary region

**Locally Redundant Storage (LRS)**

* Copies data synchronously three times within a single **physical** location (datacenter) in the **primary** region
  + Write requests occur synchronously
    - Write operation is successful **after** the data is written three times
* Protects against server rack & drive failures
  + Does **not** protect against datacenter fires or floods
    - Important: all data replicated using LRS may be unrecoverable if affected by fire/flood
    - **Microsoft recommends ZRS, GRS, or GZRS to mitigate this risk**
* Not recommended for apps needing high availability or durability
* Provides at least 99.999999999% (11 nines) durability of objects over a given year
* Lowest-cost & durability replication option

LRS might be the right choice for:

* Apps that stores data which can be reconstructed easily if data loss occurs
* Apps are restricted to replicating data within a country/region due to data governance requirements
  + **Important: some paired regions will replicate data to another country/region (geo-replication)**
* Apps using unmanaged disks
  + Unmanaged disks *can* use GRS, however, it isn't recommended due to potential issues

**Zone Redundant Storage (ZRS)**

* Copies data synchronously **across three Azure availability zones** in the **primary**region
  + Each Availability Zone is a separate **physical** location with its own independent power, networking, and cooling
    - Write requests occur synchronously
      * Write operation is successful **after** the data is written across **all three** availability zones
        + Note: if a zone is temporarily unavailable, the operation will return successfully after the data is written to **all**availability zones
* Microsoft recommends ZRS for apps that require high availability, while also replicating to a secondary region
  + Recommended to use **ZRS in the primary region for Azure Data Lake Storage Gen2 workloads**
  + Recommended for **restricting the replication of data to a specific country/region** to meet any data governance requirements
    - Recommended to use for **Azure Files workloads** - if a zone becomes unavailable, remounting the file shares is **not** required
* Provides at least 99.9999999999% (12 nines) durability over a given year
* Data is accessible (read and write) even if a zone becomes unavailable
  + Azure will undertake networking updates, like DNS repointing, if a zone becomes unavailable
* Provides excellent performance, low latency, and resiliency for data if it's temporarily unavailable
  + Does **not** protect against regional disaster where multiple zones are permanently impacted.
    - Important: Microsoft recommends using GZRS to mitigate against this risk as it uses ZRS in the **primary** region while also geo-replicating data to a **secondary region**

It's best to follow practices for transient fault handling, such as implementing retry policies with exponential back-off, when designing apps using ZRS.

**Important: Archive tier for Blob Storage is not currently supported for ZRS, GZRS, or RA-GZRS**

* Unmanaged disks do **not** support ZRS or GZRS

Redundnacy in Secondary region: data is copied to a secondary region, making the data durable even in the event of a **complete regional outage** or diaster that makes the primary region unrecoverable

**Geo-redundant Storage (GRS)**

* Copies data **synchronously three** times within a single, physical location in the **primary region** using LRS
  + Then, data is copied **asynchronously**to a single, physical location in the secondary region where the data will be copied **synchronously three times**using LRS

**Geo-zone-redundant Storage (GZRS)**

* Copies data **synchronously across three Azure Availability Zones** in the **primary region** using ZRS
  + Then, data is copied **asynchronously**to a single, physical location in the secondary region where the data will be copied **synchronously three times**using LRS

Important: the main difference between GRS and GZRS is how data is replicated within the **primary** region.

* Data is **always** replicated synchronously three times using LRS within the secondary region.
  + Replication using LRS in the secondary region protects data against **hardware failures**.

Important: with GRS and GZRS, data within the secondary region is **not** available for **read or write access** UNLESS there is a **failover to the primary region**.

* To gain read access to the secondary region, configure the storage account to use Read-Access Geo-redundant Storage (RA-GRS) or Read-Access Geo-zone-redundant Storage (RA-GZRS).
* If the primary region becomes unavailable, there is the option to choose to fail over to the secondary region.
  + Once the failover completes, the secondary region will become the primary region, restoring the ability to read and write data: [read more about disaster recovery](https://learn.microsoft.com/en-us/azure/storage/common/storage-disaster-recovery-guidance).

Important: since data is replicated to the secondary region **asynchronously**, failure that affects the primary region may result in **data loss** if the region cannot be recovered.

* The Recovery Point Objective (RPO) is the interval between the most recent writes to the primary region and the last write to the secondary region.
  + RPO indicates the point in time to which data can be recovered.
  + Azure Storage usually has an RPO of less than 15mins, however, there is **not** an SLA currently for how long it takes to replicate data to the secondary region.

### **Storage encryption**

Azure Storage Encryption is enabled by default on **all** storage accounts.

* This includes Resource Manager & Classic Storage Accounts.

Benefits of Storage Service Encryption (SSE):

* Automatically encrypts & decrypts data using 256-bit AES encryption
* Protects data for security & compliance
* Enabled for **all** storage accounts, whether new or exisiting
* Options for managing encryption keys
  + Microsoft Managed Keys: Microsoft rotates the keys on behalf of the customer.
  + Customer Managed Keys: bring your own keys and store them in an Azure Key vault, which is a key management system (PaaS).

### **Storage tools**

Azure Storage Explorer

* Access multiple accounts & subscriptions
* Create, delete, view, and modify storage resources
* View & edit Blobs, Queues, Tables, Files, Cosmos DB, and Data Lake
* Obtain SAS tokens/keys
* Supported OS: Windows, macOS, and Linux

Azure Import/Export Services

* Import Services moves data to Azure blob storage **or** files
* Export Services **only** moves data to Azure blob storage

AzCopy

* CLI tool
* Supported OS: Windows, macOS, and Linux
* Designed to copy data to and from blob, file, and table storage
* Authentication options: - Active Directory - SAS Tokens

## **Configure Azure Files and Azure Blob Storage**

* Create and configure a file share in Azure Storage
* Create and configure a container in Blob Storage
* Configure storage tiers
* Configure snapshots and soft delete for Azure Files
* Configure blob lifecycle management
* Configure blob versioning

### **Immutable Storage - Blob Storage**

Immutable storage for Blob Storage allows users to store business-critical data in a WORM (Write Once, Read Many) state.

* While in this state, data cannot be modified or deleted for a user specified time.
  + Important: Immutability policies protects your data from overwrites and deletes.

There are two types of immutability policies:

* Time-based retention policies: policies are set to store data for a specific timeframe.
  + Data can be created and read, but cannot be edited or deleted.
  + When the retention period expires, objects can be deleted but **cannot** be overwritten.
* Legal hold policies: these store the data until the legal hold is explicitly cleared.

# Deploy and manage Azure compute resources

## **Automate deployment of resources by using Azure Resource Manager (ARM) templates or Bicep files**

* Interpret an ARM template or a Bicep file
* Modify an existing ARM template
* Modify an existing Bicep file
* Deploy resources by using an ARM template or a Bicep file
* Export a deployment as an ARM template or compile a deployment as a Bicep file

### **Azure Resource Manager (ARM)**

ARM provides the ability to work with the resources in your solution as a group instead of individual components.

* Deploy, update, or delete all resources for solutions in a single operation
* Use templates for deployment
* Templates can work for testing, staging, and production environments
* Provides security, auditing, and tagging to assist with managing resources after deployment
* Provides the ability to monitor all resources for your solution
* Repeatedly deploy solutions throughout the development lifecycle while ensuring they're deployed consistently
* Templates are declarative and can be used to manage the infrastructure instead of using scripts
* RBAC can be applied to all service within the resource group since it's natively integrated into the management platform
* Tags can help logically organize resources within the subscription
* The organization's billing can be clarified by viewing the costs for groups of resources using the same tags

ARM provides a way to consistently manage resources by performing tasks through:

* Azure PowerShell
* Azure CLI
* The Azure portal
* REST API
* Client SDKs

Take advantage of ARM while working with solutions by:

* Defining all deployment and config steps using the template(s). No manual steps should be required to setup solutions
* Running commands like starting/stopping an app or machine to manage resources
* Organizing resources with the same lifecycle using resource groups
  + Use tags to organize additional resources

### **ARM Templates**

ARM Templates are JSON documents, written in a declarative syntax.

* Declarative syntax is what powers declarative automation, which is the automated deployment of resources.

Benefits include:

* Improve consistency: the structure, format, and expresses in the template will remain the same regardless of what tools are used to deploy the ARM template(s).
* Link templates together: allows the templates to become modular.
* Save time & prevent errors: manually creating & connecting resources is time consuming and leave room for errors.
* Easily express complex deployments: multiple resources can be deployed in the correct order using templates.
  + ARM will map out each resource and the dependencies to ensure dependent resources are created and deployed first.
* Express requirements through code: templates are a type of IaaS and can be shared, tested, and versioned.
  + Can be used with Git for version control.
* Promote reuse: templates can require params that are filled out when it runs.
  + Ex: domain name, username or password, etc.
* Simplify orchestration: instead of deploying each resource independently, all of your resources can be deployed by deploying the template.

### **ARM Template Schema**

JSON documents are a collect of key-value pairs.

* Each key is a string and the values can be:
  + A string
  + List of values
  + Number
  + Boolean expression
  + An object (aka a collection of other key-value pairs)
* $schema: the location of the JSON schema file that describes the version of the template language.
  + Required
* contentVersion: the version of the template. This value is used to document major changes in the template & can be used to identify the proper template is being utilized.
  + Any value can be provided for this.
  + Required
* parameters: values that are provided when deployment is executed to customize resource deployment.
  + Not required - Important: each template is limited to 256 params; reduce the number by using objects that contain multiple properties.
* variables: values that are used as JSON fragments in the template which simplify the template language expressions.
  + Not required
* functions: user-defined functions available within the template
  + Not required
* resources: the types of resource that gets deployed, or updated, in a resource group.
  + Required
* outputs: values that are returned **after**deployment
  + Not required

### **Bicep Templates**

Azure Bicep is a domain-specific language (DSL) and can be used instead of JSON to develop ARM templates.

* Like ARM templates, it uses declarative syntax to deploy resources and supports code reuse.
* Additionally, it provides reliable type safety.

Bicep syntax:

* Reduces complexity
* Improves the development experience

Note: Bicep is a transparent abstraction over ARM templates and don't lose any of the capabilities of JSON templates.

When deploying a resource, or multiple resources, Bicep will convert the template into a JSON template.

* Transpilation: the process of converting source code to another language.

Benefits of using Bicep over JSON templates:

* Simple syntax: provides a simpler syntax for writing templates which makes reading and writing Bicep templates easier.
  + Params and variables can be referenced directly.
  + String interpolation is used to combine values for names and other items, instead of concatenation.
  + Properties of a resource can be referenced directly by using the symbolic name vs complex reference statements.
* Automatic dependency management: Bicep autodetects dependences between resources which reduces some of the work required to write templates.
* Type validation & IntelliSense: VS Code offers a Bicep extension which provides rich validation and IntelliSense for **all** Azure resource type API definitions, which makes writing templates easier.
* Modules: complex templates can be broken down into smaller module files, which can be referenced in the main template.
  + Provides easier management as well as making the template writing process easier.

## **Create and configure virtual machines**

* Create a virtual machine
* Configure Azure Disk Encryption
* Move a virtual machine to another resource group, subscription, or region
* Manage virtual machine sizes
* Manage virtual machine disks
* Deploy virtual machines to availability zones and availability sets
* Deploy and configure an Azure Virtual Machine Scale Sets

### **Azure Virtual Machines**

Working with VMs allows you to have more control over the installed software and configuration settings.

Characteristics of Azure VMs include:

* Based on the infrastructure as a service (IaaS) model, which is an instant computing infrastructre that's provisioned and managed over the web.
* VMs provide their own OS, storage, and networking capabilities, and can run a wide range of apps.
* VMs can be quickly scaled up and down on demand, allowing you to pay for what you use.
* Multiple VMs can be implemented, and each machine can be configured with different software and config settings to support different operations (deployment, testing, and deployment environments).
* Microsoft and the customer share responsibility associated with configuring and maintaining VMs.

Before using IaaS and VMs, you should consider the following:

* Test and development: quickly setup and tear down test & dev environments to push apps out faster.
* Website hosting: using IaaS and VMs can make running websites less expensive than traditional hosting.
* High-performance computing (HPC): HPC can solve complex problems that involve millions of calucations or variables. VMs enable this on clusters, computer grids, or supercomputers, providing the ability to support various scenarios.
* Storage, backup, and recovery: VMs avoid expenses associated with storage and storage management. IaaS provides a simplified approach to handling unpredictable demand, as well as backup and recovery systems.
* Extended datacenters: VMs in Azure can help add capacity to an organizations datacenter, without requiring addition cost for physical hardware and additional maintenance.
* Big data analysis: data mining allows you to discover hidden patterns that require a large amount of processing power, provided by IaaS in a more economical approach.

### **Planning VM Configuration**

Following a checklist will help you plan to configure a VM:

* Network configuration: VNets are used to provide private connectivity between your VMs and other services in Azure.
  + Resources on the same VNet can access one another; by default, any resources outside the VNet aren't able to connect to those within the VNet.
  + Additional configuration is required to the network before external services can be accessed, including any on-prem servers.
    - Note: If your organizations private network is going to be connected to the services in Azure, make sure to consider the topology before creating VMs.
* Name for the VM: the name of the VM will be configured as part of the OS.
  + Windows VMs allow names up to 15 characters long
  + Linux VMs allow names up to 64 characters long
    - Note: make sure to pick a naming convention that's meaningful and standardized.
      * Ex: produsc-webvm01 would be used for the first production web server in the US South Central location.
* Location for the VM: the region in which you create your VM will be where the resources are allocated.
  + Choosing a regional location allows your VM to be close to your users.
    - Note: the location of the VM can improve performance and is an important decision to ensure you meet legal, tax, and/or compliance requirements.
  + Location can limit your options for hardware and configurations.
  + Location can also determine the pricing - some regions will be more expensive than others.
* Size of the VM: depending on the type of workload the VM will be running will determine the memory and storage sizes your machine will require.
  + Azure allows you to resize VMs when the size no longer meets your needs if the hardware configuration is allowed in the new size.
  + This provides an agile and elastic approach to VM management.
    - Note: you can select any size available in your region when you stop & deallocate the VM. However, you should use caution resizing production machines as resizing might require a reboot or change the IP address, causing a temp outage.
* Review the pricing model to estimate costs: subscriptions are billed two charges for each VM; compute and storage. These costs being separate allow you to scale them and pay for only what you need.
  + Compute: paid on a per-hour basis and billed on a per-minute basis.
    - Consumption-based means you pay for the compute capacity per second. This option allows you to increase or decrease the capcity on demand, allowing you to start or stop as needed.
      * This option is best if your organization runs apps with unpredictable or short-term workloads that cannot be interrupted, such as quick tests or developing apps in VMs.
    - Reserved VM Instances (RI) provides an option to purchase a VM for one or three years, in a specific region. While this option requires a commitment made up front, you save up to 72% compared to the pay-as-you-go pricing model. - This option is best if the VM needs to run continuously OR if you need a predictable budget, if you can commit to using the VM for a minimum of one year. - RIs can be exchanged or returned with an early term fee.
  + Storage: charged separately for the Azure Storage used by the VM.
    - Status of the machine doesn't determine the charges incurred.
* Select an OS to use: pick between different versions of Windows or Linux.
  + The cost of the OS license is included in the price.
  + Use the Azure Marketplace to search for more than just base OS images. This allows you to install images that have the OS and additional software.
    - Ex: A Linux server, Apache web server, MySQL database, and PHP are included in an image stack.
  + You can also create your own disk image if you cannot find one that suits your needs.
    - Note: Azure only supports 64-bit OS.
* Identify the Azure Storage to use: Azure Managed Disks handle the storage account creation & management for you.
  + Disk size & performance (standard or premium) can be selected.
  + Adding disks or scaling up/down is possible.
* Determine VM storage: all Azure VMs have *at least* two disks: an OS and a temporary disk. However, they can have one or more data disks.
  + All disks are stored as Virtual Hard Disks (VHDs), which are like physical disks in on-prem servers, just virtualized.
    - OS disk: every VM will have one OS disk attached.
      * These disks contain the pre-installed OS and are registered as a Serial Advanced Technology Attachedment (SATA) drive.
      * OS disks will be labled as the C: drive by default.
    - Temp disk: data on these disks could be lost during maintenance or upon deployment, however, it should persist upon a normal reboot of the VM.
      * Windows: by default, this drive will be labeled the D: drive and is used to store the pagefil.sys file.
      * Linux: this drive is usually /dev/sdb and is formatted & mounted to /mnt by the Azure Linux Agent.
        + Important: data on the temp drive should not be critical to the system as it could be lost upon moving to a new host.
    - Data disks: these are managed disks and used to store app data, or any other data that's needed to be kept.
      * Registered as SCSI drives and are able to be labeled a letter of your choice.
        + Note: the size of the VM determines how many data disks can be attached and the kind of storage that can be used to host these disks.

### **Storage considerations**

* Azure Premium Storage: gain high-performance, low-latency disk support for VMs with input/output (I/O)-intensive workloads.
  + Premium Storage stored data on SSDs.
  + Existing VM disks can be migrated to Premium Storage.
* Multiple Storage disks: using multiple disks allows your apps up to 256 TB of storage space per VM.
  + Premium Storage allows your apps to achieve 80,000 I/O operations per second (IOPS) per VM & a disk throughput up to 2,000 MB/s per VM.
  + Read operations yield low latencies with Premium Storage.
* Managed disks: Azure Managed Disks are VHDs and are stored as page blobs which are a random IO storage object.
  + These disks are considered *managed*because they are an abstraction over page blobs, blob containers, and Azure storage accounts.
  + You provision these disks while Azure handles creating and managing them.
  + These types of disks are available in Ultra SSD, Premium SSD, Standard SSD, and Standard HDD.
    - Note: These disks are required for the single instance VM SLA.
* Migrating to Premium Storage: Microsoft recommends migrating VM disks that require high IOPS to Premium Storage for the best performance.
  + If the disk doesn't require high IOPS, reduce your cost by keeping it in standard storage.

### **Creating VMs in Azure portal**

The first thing to decide is which image to use while creating a VM in the Azure portal. The portal will guide you through the config process while creating the machine image.

* Basics tab: contains the admin account, project details, and inbound port rules.
* Disks tab: select the OS disk type & data disks.
* Networking tab: create VNets and load balancing.
* Management tab: enable auto-shutdown and configure backup details.
* Advanced tab: configure scripts, agents, or VM extensions.
* Monitoring & Tabs tabs: additional settings are available.

### **Connecting to VMs**

You can connect to your VMs in Azure in multiple ways:

* Azure Bastion
* SSH
* RDP
* Cloud Shell

### **Windows-based VMs**

Connecting to a Windows VM hosted in Azure will be done using the Remote Desktop app (uses RDP).

* Any version of Windows that's supported will have a GUI session.
  + The VMs IP address and port number will be needed to connect with this method. Azure will then allow you to download a RDP file to use to connect.

### **Linux-based VMs**

Connecting to a Linux VM hosted in Azure can be done using an SSH client.

* You can reuse a single public-private key pair to access VMs and services, depending on the org's security policies.
  + WSL, Terminal, or Shell can be used as SSH clients to connect.
  + The path to your private key file will be required. This key remains on your system and needs to be protected.
  + The public key will be placed on the Linux VM, or another service using public-key cryptography.

### **Azure Bastion**

Note: this is a platform managed PaaS service.

Bastion provides secure, seamless RDP/SSH connectivity to all VMs in the VNet, directly over SSL.

* No IP address is required when connecting using the service.
* Azure Bastion protects VMs from exposing RDP/SSH ports externally.
  + Note: this service allows you to connect directly from Azure portal; no additional software is required.

### **Planning for Maintenance**

Availability plans for VMs need to include methods for unplanned and unexpected situations, and planned situations.

Unplanned hardware maintenance: these events happen when Azure predicts the a component associated to the physical machine is going to fail. When the platform predicts a failure, an unplanned hardware maintenance event will be issued. - Note: Azure uses Live Migration technology to migrate VMs from failing hardware to a fully functioning machine. - Live Migration will only pause the VM for a short period of time and performance *might* be impacted before or after.

Unexpected downtime: these events happen when the infrastructure or hardware for your VM fails. This could include:

* Local network failures
* Rack level failures
* Local disk failures
  + Note: Azure will automatically migrate the VM to a healthy physical machine within the same datacenter when this occurs.

Regularly scheduled maintenance: these events are regular updates Microsoft makes to the Azure platform to help improve reliability, security, and performance of the infrastructure VMs run on.

* It's important to remember that *most* of these will not impact any VM or service running in Azure.
  + Note: Microsoft does not automatically update VM OS or other software. You are fully responsible for these updates. Only the underlying software is updated periodically, as well as the hardware.

### **Availability sets**

There are several things to remember about availability sets, including but not limited to:

* A VM and an availability set can be created at the same time
  + Note: VMs can only be added to an availability set when the VM is created. Changing the availability set requires deleting the VM and then recreating it.
* Availability sets can be built using the following options:
  + Azure portal
  + ARM templates
  + Scripting
  + API tools
* All VMs within an availability set should perform the same set of functions
* All VMs within an availability set should have the same software installed
* VMs in an availability set will run across multiple servers, racks, storage units, and switches.
  + Note: if hardware or Azure software fails, only a subset of the VMs in the set are impacted and the app will continue to be available.
* SLAs are provided by Microsoft for VMs and availability sets.

*Important: Availability sets won't protect the apps from OS or app failures.*

When using availability sets, consider the following:

* Redundancy
* Load balancing: for high availability & network performance
* Separation of application tiers: to help mitigate single points of failure
* Managed disks: for block-level storage

An availability set is a logical grouping of virtual machines hosting your application. These consist of update domains (UDs) and fault domains (FDs). Each VM you create will have a UD and FD that's associated with it.

* Note: you can have up to 3 FDs and 20 UDs.
  + The default number of UDs is 5. If you create more than 5 VMs, the 6th VM will be placed on the 1st UD, the 7th on the 2nd UD, and so on.

Deploying multiple instances of your app will help you achieve High Availability (HA), which removes the risk of a single point of failure.

FDs represent a set of VMs that have the same network switch, A/C, and power.

* Note: you can only configure up to 3 FDs, which is the default value when setting up availability sets.

UDs represent a group of VMs and the host, which can be updated & rebooted at the same time.

* Note: only one UD will be rebooted at a time during scheduled maintenance, such as patching, firmware updates, etc.
  + While one is being rebooted, 30mins is required before the next scheduled task will begin on another domain.

Creating availability sets can be done using:

* Azure portal
* Azure PowerShell
* Azure CLI

Information required to create availability sets:

* Subscription
* Resource group
* Name
* Region
* Number of FDs
* Number of UDs
* Disk alignment (use of managed disks)

### **Fault domains and Update domains**

VM Availability sets implement two node concepts to achieve high availability and fault tolerance:

Update domains: these are a group of nodes that get upgraded together during service upgrades and allow Azure to incrementally upgrade across a deployment.

* Can be updated and rebooted at the same time
* Only one update domain is rebooted at a time during planned maintenance
* There are 5 UDs by default
* Up to 20 UDs can be configured

Fault domains: these are a group of nodes representing physical units of failure.

* Two FDs work together to prevent hardware failure, network outages, power interruptions, and/or software updates.
* FDs are a group of VMs sharing the same hardware, or network switch, and sharing the same single point of failure.

### **Availability zones**

Each Azure region has unique physical locations. These locations are called zones and make up one or more datacenters that has its own power, networking, and cooling. Deploying apps to multiple zones can provide protection from datacenter failure as we can create instances in different zones within a region.

* If a zone suffers an outage, instances will be deployed in different zones to maintain availability.

When creating a VM, you can select which zone to deploy to.

*Note: Availability zones can help improve SLAs of your applications.*

* Availability zones are unique, *physical*locations within a region in Azure.
* Physical separation within regions protects against datacenter failures.
* Each zone has one, or more, datacenters equipped with power, cooling, and networking independently.
* Zone-redundant services will replicate apps and data across the availability zones, protecting against single points of failure.
* A minimum of 3 separate zones exist within enabled regions.

There are two categories of services that support availability zones:

Zonal services: these pin each resource to a zone

* Azure VMs
* Managed disks
* Standard IP addresses

Zone-redundant services: these replicate automatically across *all* zones

* Azure SQL DB
* Zone-redundant Azure Storage

*Important: Availability zones help businesses achieve business continuity while using Azure services. However, you should build the architecture of your apps using a combo of availability zones and region pairs.*

### **Vertical and horizontal scaling**

Scalability allows throughput for a VM proportionally to the availability of the associated hardware resources. When a VM is scalable, it can handle increased requests without negatively affecting throughput or response time.

Vertical scaling (aka scale up/scale down): increases or decreases the VM size in response to a workload. This makes the VM more or less powerful.

Vertical scaling can be beneficial when:

* Services running on a VM are not heavily used on the weekends: scaling down can help decrease monthly costs
* An increase in use occurs: scaling up can help you support the demand without requiring the creation of additional VMs

Horizontal scaling (aka scale in/out): adjusts the number of VMs to support a changing workload.

Consider the following when using either scaling option:

* Limitations: horizontal scaling has fewer limitations compared to vertical scaling as vertical scaling depends on the availability of larger hardware.
  + Note: larger hardware hits upper limits quickly and availability varies by region.
  + Vertical scaling typically requires stopping the VM and restarting it, which presents limitations to accessing data or apps.
* Flexibility: horiztonal scaling is more flexible in the cloud.
* Reprovisioning: the process of removing an existing VM and replacing it with a *new*machine.

### **Azure Virtual Machine Scale Sets (VMSS)**

Virtual Machine Scale Sets are an Azure Compute resource that can be used to deploy & manage a set of identical VMs.

* All VMs are configured in the same way allowing you to have true autoscaling
* There is no need to pre-provision VMs
* VMSS provides an easier way of building large-scale services that target large compute, containerized workloads and big data
* VMSS automatically increases/decreases the number of VM instances as demand increases/decreases
  + Note: adding and removing machines can be automated or manual, or a mixed combination.

It's important to remember:

* VMSS supports up to 1,000 VM instances.
  + If you crease and upload custom VM images, you are limited to 600 VM instances.
* VMSS supports the use of Azure Load Balancer (basic layer 4 traffic distribution).
* VMSS supports the use of Azure Application Gateway (advanced layer 7 traffic distribution & SSL termination).
* All VM instances are created from the *same* base OS image & configuration.
* VMSS can be used to run multiple instances of apps.
  + If one VM has a problem, users will keep their access through other VMs with minimal interruption.
* VMSS autoscales based on the demand for your app(s).

### **Create VMSS**

Creating VMSS can be done from the Azure portal. The following information is needed:

* The number of VMS
* Allocation policies
* Orchestration mode: how VMs are managed by the scale set
  + Uniform: optimized for large scale stateless workloads with identical instances - define the VM model and Azure generates identical instances based on the model.
  + Flexible: achieve high availability at scale with identical or multiple VMs - manually create & add VMs of *any*configuration to the scale set.
* Image/base OS or app for the VM
* VM architecture:
  + Arm64
  + x64
* Run with Azure Spot discount: offers the unused Azure capacity at a discount vs pay as you go
  + Note: workloads should be tolerant to infrastructure loss as this may recall capacity
* Size: select the size to support the workload you want to run.
  + Note: this size will determine processing power, storage capacity, and memory.
  + Azure charges an hourly price based on the size and OS.

Advanced options can also be selected:

* Enable scaling beyond 100 instances:
  + Selecting No means the VMSS implementation will be limited to one placement group with a max capacity of 100.
  + Selecting Yes means the implementation can span multiple placement groups with a max capacity of 1,000. However, this will change the availability characteristics.
* Spreading algorithm: Microsoft recommends allocating *Max spreading* to provide the optimal spreading.

### **Autoscaling**

Consider the following:

* Automatic adjusted capacity: rules can be created to define the acceptable performance for positive user experience - when the thresholds are met, the rules will adjust the capacity of the VMSS implementation.
* Scale in: reduce the cost to run the VMSS implementation - limits the number of instances that will run to provide support to the current demand.
* Scale out: automatically increase the implementation if the load on your VMs increases.
* Scheduled events: events can be scheduled to automatically increase/decrease the capacity of the implementation at specific times.
* Overhead: VMSS with autoscaling reduces management overhead to monitor and optimize the performance of your app.

Autoscaling can be enabled during the creation of VMSS.

* Note: make sure to define a min., max., and default number of VM instances to use during the process for best performance.

Several settings can be configured to enable autoscaling:

* Min/Max number of VMs
* Scale out
  + CPU threshold %: specifies the threshold to trigger the scale out rule
  + Duration in mins: defines the amount of time the Autoscale engine will look for metrics. The delay allows metrics to stabilize and avoids reacting to transient spikes.
  + Number of instances to increase by: defines the number of VMs to add to the VMSS when the scale out rule is triggered
* Scale in
  + CPU threshold %: specifies the threshold to trigger the scale in rule
  + Number of instances to decrease by: defines the number of VMs to remove from the VMSS when the scale in rule is triggered
* Scale-in policy: this feature provides a way to configure the order VMs are scaled in.

## **Provision and manage containers in the Azure portal**

* Create and manage an Azure container registry
* Provision a container by using Azure Container Instances
* Provision a container by using Azure Container Apps
* Manage sizing and scaling for containers, including Azure Container Instances and Azure Container AppsConfigure sizing and scaling for Azure Container Instances

## **Create and configure an Azure App Service**

* Provision an App Service plan
* Configure scaling for an App Service plan
* Create an App Service
* Configure certificates and TLS for an App Service
* Map an existing custom DNS name to an App Service
* Configure backup for an App Service
* Configure networking settings for an App Service
* Configure deployment slots for an App Service

### **Azure App Service plans**

Azure App Service plans define a set of compute resources for web apps to run.

* One or more application can be configured to run on the same computing resources (in the same App Service plan).
* When creating an App Service plan in a region, a set of the compute resources is created for the plan in that region. Any apps placed into the plan will run on the compute resources the plan uses.
* New apps can be added to existing plans, if the plan has the resources to handle the increased load.

App Service plans define the following settings:

* Region
* Number of VM instances
* Size of VM instances

App Service Pricing Tiers:

* Free or Shared Tier
  + Apps run by reciving CPI minutes on a shared VM instance
  + Apps cannot scale out
* Basic, Standard, Premium, or Isolated Tier
  + Apps run on *all* VM instances in the App Service plan
  + Multiple apps in the same plan will share the same VM instances
  + If there are multiple deployment slots for an app, they will all run on the same VM instances
  + Enabling diagnostic logs, running WebJobsm and performing backups use CPU cycles & memory on the same VM instances

Note: If autoscaling is configured, all apps in the plan are scaled out together based on the settings configured.

### **App Service Pricing**

There are a total of 6 categories of pricing tiers.

Free & Shared: these plans are the base tiers that run on the same VMs as other apps.

* Some apps might belong to other customers.
* Intended for dev & testing.
* No SLA.
* Metered on a per app basis.

Basic: these plans are designed for apps with lower traffic requirements and don't require advanced autoscaling or traffic management.

* Pricing is based on size & number of instances.
* Built-in network load balancing support distributes traffic across instances automatically.
* Supports Web App for Containers with Linux runtime environments.

Standard: these plans are designed to run production workloads.

* Pricing is based on size & number of instances.
* Built-in load balancing support distributes traffic automatically across instances.
* Autoscale is included.
* Supports Web App for Containers with Linux runtime environments.

Premium: these plans are designed for enhanced performance for production apps.

* Premium v2 offers Dv2-series VMs with faster processors, SSD, and double memory-to-core ratio vs the Standard tier.
* The new plan supports higher scale via increased instance count while providing the advanced capabilities of the Standard tier.
* The first gen of Premium is available to support existing customer scaling needs.

Isolated: these plans are designed for mission critical workloads that are required to run in a VNet.

* Allows apps to run in a private, dedicated environment in an Azure datacenter.
  + Note: this private environment is referred to as the App Service Environment.
* Offers Dv2-series VMs with faster processors, double memory-to-core ratio, and SSD compared to the Standard tier.
* Can scale to 100 instances, but more are available upon request.

### **App Service Scaling**

Scaling up increases the amount of CPU, memory, and disk space available.

* Provides additional features: dedicated VMs, custom domains & certificates, autoscaling, staging slots, plus more.
* Scale up by changing the pricing tier of the App Service plan where the app is.

Scale out increases the number of VM instances that run the app.

* Scale out up to 30 instances, depending on the pricing tier.
  + The Isolated tier allows you to scale out up to 100 instances.
* Scale instance count can be configured manually or automatically.

Autoscale allows you to automatically increase the scale instance count for scaling out.

* Based on predefined rules & schedules.

Scale up or down anytime by changing the pricing tier.

### **Azure App Services**

Azure App Services seamlessly brings together the components needed to create mobile backends, websites, and web APIs for any device or platform. These apps easily run and scale within both Linux and Windows environments.

Benefits of Azure App Services:

* DevOps integration: continuous integration and deployment using Azure DevOps, GitHub, Docker Hub, and BitBucket.
  + Apps can be managed in App Service using Azure PowerShell or the CLI.
* Support for various languages and frameworks: provides support for PHP, Ruby, Python, ASP.NET, Java, and Node.js.
  + PowerShell and other scripts/executables can be ran as background services.
* Serverless code: code snippets can be ran on demand
* High Availability: the Basic tier and above allow you to use built-in load balancers.
  + Note: autoscaling is available from the Standard tier and above
* Compliance & Security: Azure App Service is ISO, PCI, and SOC compliant and allows for easy other IdPs to be integrated for authentication.
  + Ex: Google, Facebook, Twitter, Microsoft.
* VS integration: dedicated tools in Visual Studio are offered to help publish apps.
* Marketplace app templates: lists of app templates are available within the Azure Marketplace.
  + Ex: Joomla, Drupal, and WordPress.
* API and mobile features: features like CORS support for RESTful API, offline data syncing, and push notifications are offered using Azure App Services.

### **Creating Apps with App Service**

Configuration settings to create an app with App Service:

* Name: the name must be unique as it identifies the app and its location in Azure.
  + Custom domains can be mapped if preferred.
* Publish: App Service hosts the app as a Docker container or as code.
* Runtime stack: multiple choices are available, along with various versions of PHP, Python, Ruby, Node.js, .NET Framework, and .NET Core.
  + Linux apps and custom container apps can have an optional start-up file/command configured.
* OS: Linux or Windows
* Region: the region will affect the App Service plans that are available for your app.
* App Service plan: the app needs to be associated with an Azure App Service plan for available resources, capacity, and features.
  + Pricing tiers will depend on what's available in the region selected.

Once the app is created, additional configuration settings will become available.

* App deployment
* Path mappings
* Always On: your app can stay loaded regardless of the volume of traffic.
  + Required for continuous WebJobs, or WebJobs that are triggered by using a CRON expression.
* ARR affinity: you can ensure the app client is routed to the same instance for the life of the session while in a multi-instance deployment.
* Connection strings: these strings are encrypted at rest & are transmitted over an encrypted channel.

### **Security with Azure App Service**

App Service offers built-in authn & authz support.

* Sign in users and access data by writing minimal or no code in your apps, whether web app, mobile backend, Azure Functions, or API.
  + Note: you don't have to use App Service for authentication and authorization.

Important to know:

* The authn and authz security module runs in the **same** env as your app code, but separately.
* The security module is configured using app settings - no changes to your app code, no SDKs, or specific languages are needed
* Incoming HTTP requests pass through the module after the security module is enabled, before being handled by the app code.
* The module handles the following tasks for your app:
  + Authentication of users with a specified provider
  + Validates, stores, & refreshs tokens
  + Manages the authenticated session
  + Injects identity info into request headers

You can choose to do the following:

* Allow anonymous requests: provides flexibility to handle anonymous requests and allows you to have multiple IdPs available to users.
* Logging and tracing: authentication and authorization traces can be viewed directly from the log files.
* Allow only authenticated requests: all anonymous requests will be redirected to the provider of your choice. No authentication code will need to be written

### **Custom domain names**

Azure will assign newly created web apps to the subdomain azurewebsites.net and will assign a virtual IP address for the app.

* For production apps, you can show your users a custom domain name.

Note: to map a custom domain, you need to have a paid tier for your Azure App Service plan.

Steps to create a custom domain:

1. Reserve your domain name: you can purchase one in the Azure portal, which is the easiest way to set it up.
2. Create DNS records to map the domain to the web app: A and CNAME records.
   * A record will map the domain to an IP address.
   * CNAME record will map the domain to another domain.
3. Enable the custom domain: use the Azure portal to validate the domain and add to your web app.
   * Note: always test the domain before publishing

# 

# Configure and manage virtual networking

## **Configure virtual networks**

* Create and configure virtual networks and subnets
* Create and configure virtual network peering
* Configure private and public IP addresses
* Configure user-defined network routes
* Configure Azure DNS

### **Virtual networks**

Virtual networks represent your network in the cloud. Working with the VNet service within Azure is similar to working with a traditional network on-prem, but there are some added benefits:

* Availability
* Network isolation
* Scalability

VNets allow us to:

* Link other VNets within either the same or different regions for private communication.
* Allow VMs to connect securely using the internet.
* Create private connections between VMs and other Azure services.
* Extend on-prem architecture to the cloud, providing hybrid solutions.

### **Address space**

We manage address spaces, both on-prem and in the cloud, with the CIDR block. Azure, however, uses both private and public addresses that must be specified upon creating a VNet.

* IP addresses assigned to your org's resources will be assigned from the address space.

Best practice is to ensure your address space doesn't overlap with others in the org, otherwise, conflicts will arise upon connecting VNets to eachother or while establishing a hybrid connection.

### **Subnets**

Subnetting helps segment addresses, or different workloads, while enhancing the address allocation in an efficient manner. Once the VNet has been segmented, we can deploy resources to specific subnets.

* Note: this is where we can configure network security groups (NSGs).

We can determine the CIDR block depending on how many hosts we want deployed to the subnet.

* Note: subnets should not cover the entire address space of the VNet, we will want to plan ahead by reserving some IP addresses to use in the future.

### **Regions**

When it comes to VNets, regions are always scoped to one region or location, but other networks in different regions can still be connected.

* VPN gateways and VNet peering can be used to connect VNets in different locations/regions.
  + Note: the region will be chosen during the creation of a VNet and any VMs deployed to the VNet will be in the same region as the VNet.
  + If we need to move a VNet to another region, Azure Resource Mover will need to be used.

## **Configure secure access to virtual networks**

* Create and configure network security groups (NSGs) and application security groups (ASGs)
* Evaluate effective security rules
* Implement Azure Bastion
* Configure service endpoints
* Configure private endpoints

## **Configure load balancing**

* Configure Azure Application Gateway
* Configure an internal or public load balancer
* Troubleshoot load balancing

### **Availability sets**

An availability set is a logical grouping of virtual machines hosting your application. These consist of update domains (UDs) and fault domains (FDs). Each VM you create will have a UD and FD that's associated with it.

* Note: you can have up to 3 FDs and 20 UDs.
  + The default number of UDs is 5. If you create more than 5 VMs, the 6th VM will be placed on the 1st UD, the 7th on the 2nd UD, and so on.

Deploying multiple instances of your app will help you achieve High Availability (HA), which removes the risk of a single point of failure.

FDs represent a set of VMs that have the same network switch, A/C, and power.

* Note: you can only configure up to 3 FDs, which is the default value when setting up availability sets.

UDs represent a group of VMs and the host, which can be updated & rebooted at the same time.

* Note: only one UD will be rebooted at a time during scheduled maintenanc, such as patching, firmware updates, etc.
  + While one is being rebooted, 30mins is required before the next scheduled task will begin on another domain.

Creating availability sets can be done using:

* Azure portal
* Azure PowerShell
* Azure CLI

Information required to create availability sets:

* Subscription
* Resource group
* Name
* Region
* Number of FDs
* Number of UDs
* Disk alignment (use of managed disks)

### **Availability zones**

Each Azure region has unique physical locations. These locations are called zones and make up one or more datacenters that has its own power, networking, and cooling. Deploying apps to multiple zones can provide protection from datacenter failure as we can create instances in different zones within a region.

* If a zone suffers an outage, instances will be deployed in different zones to maintain availability.

When creating a VM, you can select which zone to deploy to.

*Note: Availability zones can help improve SLAs of your applications.*

### **Azure Load Balancer**

Azure Load Balancer is a network load balancer, and operates at layer 4 of the OSI model.

To distribute incoming requests across multiple instances of our apps we can use this service. Our app can be deployed to VMs or Virtual Machine Scale Sets (VMSS). It allows us to ensure high availability (HA) is achieved by spreading the requests across multiple instances.

* Inbound and outbound traffic is supported.
* Relies on load balancing rules & health probes to distribute traffic to backend servers. - Load balancing rules will determine how the traffic is distributed across backend servers. - Health probes will ensure the backend server is capable of handling the request(s). However, if the health cannot be determined, the requests will not be distributed to the server.

### **Types of load balancers:**

Public

* These will have a public IP address & internet facing.
* The IP address and port number get mapped to the private IP and port number of VMs in the backend pool.
  + Note: load balancer rules allow us to configure port numbers and handle different types of traffic.

Internal

* These will have a private IP address for *all*communication and will *not* be exposed to the internet.
* The IP address can only be reached by resources in the same VNet, from on-prem using VPN, or through peered networks.

## **Monitor virtual networking**

* Monitor on-premises connectivity
* Configure and use Azure Monitor for networks
* Use Azure Network Watcher
* Troubleshoot external networking
* Troubleshoot virtual network connectivity

# 

# Monitor and maintain Azure resources

## Monitor resources by using Azure Monitor

* Configure and interpret metrics
* Configure Azure Monitor Logs
* Query and analyze logs
* Set up alerts and actions
* Configure monitoring of VMs, storage accounts, and networks by using VM insights

## **Azure Monitor Alerts**

Azure Monitor captures logs and metrics telemetry data, enabling you to analyze and identify potential issues.

* Configure Azure Monitor to capture data for Azure services, resources, and apps.
* An alert consists of alert rules that mix the settings and conditions you want to monitor:
  + Conditions to match
  + Signals to gather from your resources
  + Resources you wish to monitor
* An alert rule specifies action groups to fulfill responsive steps when an alert is triggered - sending notifications to admins.

Azure Monitor can be accessed from the Azure portal.

Alert types:

* Metric alerts
* Log alerts
* Activity log events
* Smart detection alerts

Alert states:

* New: open and not in review
  + This is the only state that is set by the system, the other changes must be made by an admin.
* Acknowledged: in review, work is in progress
* Closed: issue is complete

Stateless alerts

* Triggered each time an alert rule condition matches the data analyzed, even if the same alert exists.

Stateful alerts

* Triggered when rule conditions match the data and the same alert does **not** exist.

## **Implement backup and recovery**

* Create an Azure Recovery Services vault
* Create an Azure Backup vault
* Create and configure backup policy
* Perform backup and restore operations by using Azure Backup
* Configure Azure Site Recovery for Azure resources
* Perform failover to a secondary region by using Azure Site Recovery
* Configure and review backup reports

### **Azure Backup**

This service can be used to backup, protect, and restore data in Microsoft's cloud. It replaces any on-prem backup solution with a solution in the cloud, providing reliability, security, and provides cost benefits.

* Agents are available to download and deploy. The agent you choose will depend on what you want to protect.
* All agents, regardless of data residing on-prem or in the cloud, can be used to backup to a Recovery Services vault.

Benefits of using Azure Backup include:

* Keeping data secure using an encryption passphrase locally.
  + Note: this passphrase is *never*transmitted or stored in Azure
* Unlimited data transfer without added fees. Data isn't limited inbound or outbound (data transferred from a Recovery Services vault during a restore).
  + Note: initial backups offline using the import/export feature for large amounts of data will be associated with a cost for inbound data.
* App-consistent backups, meaning a recovery point has all data needed to restore the backup. This ensures there isn't a need for additional fixes to restore your data, saving you time getting back to a running state.
* Automatic storage management eliminates the need to implement on-prem storage for hybrid environments. Azure Backup will handle the allocation and manage the storage, and only requires orgs to pay for what they use.
* Offload on-prem backups to the cloud without any complex backup solutions being needed.
* Short term and long term data retention as there is no limit to the timeframe data can stay in the vault. However, Azure Backup does have a limit of 9,999 recovery points/protected instance.
* Various replication types for storage and high availability.
  + Geo-redundant storage (GRS): this is the default and recommended to use for replication. It replicates the data to another, secondary region far away.
    - Note: this method is more expensive than LRS, however, it also offers a higher level of durability in the event of a regional outage.
  + Locally redundant storage (LRS): data will be replicated 3 times in a storage scale unit within a datacenter. This method is less expensive than GRS and protects against hardware failures that occur within the same datacenter the data is stored in.
    - Note: ALL copies of the data will exist within the *same* region.
* Disaster recovery as on-prem data backups are stored in the cloud.
* Secure storage as you're provided the ability to encrypt data before backing up to the cloud.

### **Azure Recovery Services vault**

When Azure Backup is configured, the data gets stored inside a Recovery Services vault.

* Note: the data is usually copies of configuration settings for workstations, servers, VMs, or workloads.

Azure services that can be backed up:

* VMs (both Windows and Linux)
* Azure SQL databases
* And others

On-prem services that can be backed up:

* Windows server
* System Center DPM
* Azure Backup server
* And others

Recovery Services vaults allow you to organize backup data while reducing the management overhead needed.

* Note: it can help track, audit, manage, and organize multiple servers from a centralize dashboard.

The following information is needed to create a vault:

* Subscription
* Resource group
* Vault name
* Region
  + Important: you *cannot* backup VMs that are in different regions.
  + Additionally, the vault will be created by default with georedundant storage. Change this before backing up data to optimize cost

### **Backup Center**

Provides a unified management experience within Azure, allowing companies to monitor, govern, analyze, and operate backups at scale.

Benefits of Backup Center include:

* Efficiently managing backups across multiple workload types, subscriptions, regions, tenants, and vaults.
* Backup center uses Workbooks from Azure Monitor/Monitor Logs (Log Analytics) to provide detailed reports on your backups.
* Admins can filter views by specifc datasource properties like the datasource subscription, resource group, and tag(s).
* Support is available in Azure VM backup (SQL, SAP HANA), Azure files backup, Blob storage backup, and Azure managed disks backup, as well as Azure database for postgreSQL server backup.
* Backup center uses the Azure Policy to govern backups.
* Community resources can also be discovered within Backup center.

# 

# Azure Miscellaneous Notes

## **Azure Portal**

* Access the portal at portal.azure.com

## **Azure Cloud Shell**

* Allows users to choose between bash and PowerShell to manage Azure resources.
* Is temporary and requires a new or existing Azure File share to be mounted.
* Offers an integrated graphical text editor (open-source Monaco Editor).
* Automatically authenticates for easy access to Azure resources.
* Sessions time out after 20 mins of inactivity.
* Runs on a temporary host provided on a per-session, per-user basis.
* Requires a resource group, storage account, and Azure File share.
* Uses the same Azure file share for bash and PowerShell.
* Assigned to one machine per user account.
* Persists $HOME using a 5-GB image held in your file share.
* Permissions are set as a regular Linux user in Bash.

## **Azure PowerShell**

* Available in the cloud using Azure Cloud Shell or locally using PowerShell for Windows, Linux, or macOS.
* Can be used in two modes, either locally, or in the cloud:
  + Interactive: manually enter each command one at a time.
  + Scripting: execute a script containing multiple commands.

Note: The base PS product comes in two variants:

* Windows PowerShell
* PowerShell 7.x
  + Can be installed on Windows, macOS, and Linux.

Important: to install PowerShell on macOS and Linux, you will use a package manager (HomeBrew on macOS).

* The package manager to use on Linux will depend on the distro you use.
  + Ubuntu & Debian: apt-get
  + Red Hat & CentOS: yum
  + OpenSUSE: zypper
  + Fedora: dnf

### **Az Module**

Allows you to work with the following Azure features:

* Containers
* DNS
* Event Hub
* ML
* Resource groups
* Storage
* VMs
* and more.

This module replaced the AzureRM module in December 2018, however, it ships with backwards compatibility with the AzureRM module.

## **Azure CLI**

* Used locally instead of in a web browser - available on Linux, macOS, and Windows through a terminal, command-line prompt, or script.
* Can be used in two modes, either locally, or in the cloud:
  + Interactive: manually enter each command one at a time.
  + Scripting: execute a script containing multiple commands.
* Commands are structured in groups and subgroups.
  + Groups represent Azure services.
  + Subgroups divide commands into logical groupings for each service.

To find commands needed for a specific service:

## **Azure Regions**

A region is a geographical area comprising at least one datacenter, but usually multiple. Datacenters are isolated from each other in close proximity & combined nnected via low-latency networks to enable fsster and seamless communication.

* Regions offer compliance & resilency options.
* Regions offer the flexibility to deploy resources to regions that are close to an organization's customers.
* Regions ensure data residency.
* Some services are region specific or limited to some regions upon launching.
* Upon deploying resources in Azure, you will be asked to select a region in most cases.
* Azure AD, Traffic Manager, and DNS don't need a region. the region will show as Global.
* Regions are paired with another region in the same geography to create "regional pairs."

An Azure geography is an area that consists of one or more Azure regions.

Examples include, but are not limited to:

* The US: consists of several regions; East US, Central US, West US, etc.
* India
* The UK

## **Azure resource terms & additional notes**

Resource: a manageable item available through Azure.

* Examples of resources include, but are not limited to:
  + VMs
  + Storage accounts
  + Databases
  + VNets
  + Web apps

Resource group: a container that holds related resources for Azure solutions.

* All resources for the solution can be included, or only the resources you want to manage as a group. This allows you to decide how to allocate resources based on what makes sense for the organization.
  + Note: Resource groups cannot be renamed, but they can have resources from multiple regions.

Resource provider: a service that supplies resources to deploy and manage through Azure Resource Manager (ARM).

* Each resource provider offers the ability to work with the deployed resources.
* Examples of resource providers are:
  + Microsoft.Compute, this supplies the VM resource
  + Microsoft.Storage, this supplies the storage account resources
  + Microsoft.Web, this supplies web app resources
  + Microsoft.KeyVault, this provides the ability to store keys & secrets

Template: a JSON file which defines one, or more, resources to deploy to a resource group.

* Also defines any dependencies between deployed resources.
* Provides the benefit of deploying resources consistently and repeatedly.

Declarative syntax: provides the ability to create what you intend, without having to write scripts to achieve it.

* ARM templates are an example of this syntax. Within the file, you'll define the properties required for the infrastructure to be deployed in Azure.