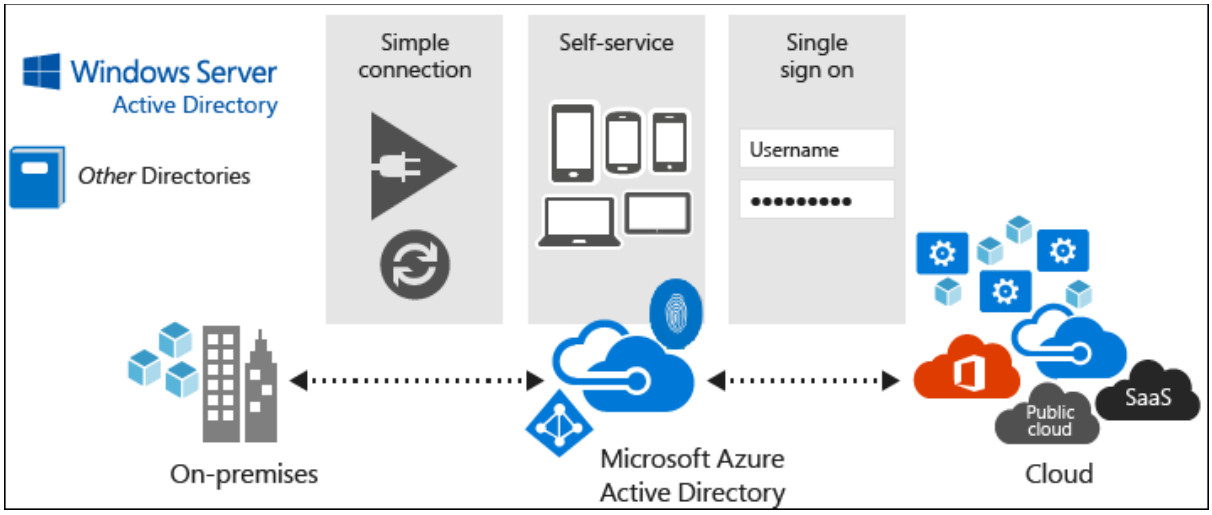
Preparation Guide for Azure Administrator - AZ-104

Microsoft Entra ID is part of the platform as a service (PaaS) offering and operates as a Microsoft-managed directory service in the cloud. It's not a part of the core infrastructure that customers own and manage, nor is it an Infrastructure as a service offering. While this implies that you have less control over its implementation, it also means that you don't have to dedicate resources to its deployment or maintenance.

With Microsoft Entra ID, you also have access to a set of features that aren't natively available in AD DS, such as support for multi-factor authentication, identity protection, and self-service password reset.

You can use Microsoft Entra ID to provide more secure access to cloud-based resources for organizations and individuals by:

- Configuring access to applications
- Configuring single sign-on (SSO) to cloud-based SaaS applications
- Managing users and groups
- Provisioning users
- Enabling federation between organizations
- Providing an identity management solution
- Identifying irregular sign-in activity
- Configuring multi-factor authentication
- Extending existing on-premises Active Directory implementations to Microsoft Entra ID
- Configuring Application Proxy for cloud and local applications
- Configuring Conditional Access for users and devices



Microsoft Entra constitutes a separate Azure service. Its most elementary form, which any new Azure subscription includes automatically, doesn't incur any extra cost and is referred to as the Free tier. If you subscribe to any Microsoft Online business services (for example, Microsoft 365 or Microsoft Intune), you automatically get Microsoft Entra ID with access to all the Free features.

Note

By default, when you create a new Azure subscription by using a Microsoft account, the subscription automatically includes a new Microsoft Entra tenant named Default Directory.

Some of the more advanced identity management features require paid versions of Microsoft Entra ID, offered in the form of Basic and Premium tiers. Some of these features are also automatically included in Microsoft Entra instances generated as part of Microsoft 365 subscriptions.

Microsoft Entra tenants

Unlike AD DS, Microsoft Entra ID is multi-tenant by design and is implemented specifically to ensure isolation between its individual directory instances. It's the world's largest multi-tenant directory, hosting over a million directory services instances, with billions of authentication requests per week. The term tenant in this context typically represents a company or organization that signed up for a subscription to a Microsoft cloud-based service such as Microsoft 365, Intune, or Azure, each of which uses Microsoft Entra ID. However, from a technical standpoint, the term tenant represents an individual Microsoft Entra instance. Within an Azure subscription, you can create multiple Microsoft Entra tenants. Having multiple Microsoft Entra tenants might be convenient if you want to test Microsoft Entra functionality in one tenant without affecting the others.

At any given time, an Azure subscription must be associated with one, and only one, Microsoft Entra tenant. This association allows you to grant permissions to resources in the Azure subscription (via RBAC) to users, groups, and applications that exist in that particular Microsoft Entra tenant.

Note

You can associate the same Microsoft Entra tenant with multiple Azure subscriptions. This allows you to use the same users, groups, and applications to manage resources across multiple Azure subscriptions.

Each Microsoft Entra tenant is assigned the default Domain Name System (DNS) domain name, consisting of a unique prefix. The prefix, derived from the name of the Microsoft account you use to create an Azure subscription or provided explicitly when creating a Microsoft Entra tenant, is followed by the **onmicrosoft.com** suffix. Adding at least one custom domain name to the same Microsoft Entra tenant is possible and common. This name utilizes the DNS domain namespace that the corresponding company or organization owns. The Microsoft Entra tenant serves as the security boundary and a container for Microsoft Entra objects such as users, groups, and applications. A single Microsoft Entra tenant can support multiple Azure subscriptions.

Microsoft Entra schema

The Microsoft Entra schema contains fewer object types than that of AD DS. Most notably, it doesn't include a definition of the computer class, although it does include the device class. The process of joining devices to Microsoft Entra differs considerably from the process of joining computers to AD DS. The Microsoft Entra schema is also easily extensible, and its extensions are fully reversible.

The lack of support for the traditional computer domain membership means that you can't use Microsoft Entra ID to manage computers or user settings by using traditional management techniques, such as Group Policy Objects (GPOs). Instead, Microsoft Entra ID and its services define a concept of modern management. Microsoft Entra ID's primary strength lies in providing directory services; storing and publishing user, device, and application data; and handling the authentication and authorization of the users, devices, and applications. The effectiveness and efficiency of these features are apparent based on existing deployments of cloud services such as Microsoft 365, which rely on Microsoft Entra ID as their identity provider and support millions of users.

Microsoft Entra ID doesn't include the organizational unit (OU) class, which means that you can't arrange its objects into a hierarchy of custom containers, which is frequently used in on-premises AD DS deployments. However, this isn't a significant shortcoming, because OUs in AD DS are used primarily for Group Policy scoping and delegation. You can accomplish equivalent arrangements by organizing objects based on their group membership.

Objects of the Application and servicePrincipal classes represent applications in Microsoft Entra ID. An object in the Application class contains an application definition and an object in the servicePrincipal class constitutes its instance in the current Microsoft Entra tenant. Separating these two sets of characteristics allows you to define an application in one tenant and use it across multiple tenants by creating a service principal object for this application in each tenant.

Microsoft Entra ID creates the service principal object when you register the corresponding application in that Microsoft Entra tenant.

Compare Microsoft Entra ID and Active Directory Domain Services

Characteristics of AD DS

AD DS is the traditional deployment of Windows Server-based Active Directory on a physical or virtual server. Although AD DS is commonly considered being primarily a directory service, it's only one component of the Windows Active Directory suite of technologies, which also includes Active Directory Certificate Services (AD CS), Active Directory Lightweight Directory Services (AD LDS), Active Directory Federation Services (AD FS), and Active Directory Rights Management Services (AD RMS).

When comparing AD DS with Microsoft Entra ID, it's important to note the following characteristics of AD DS:

- AD DS is a true directory service, with a hierarchical X.500-based structure.
- AD DS uses Domain Name System (DNS) for locating resources such as domain controllers.
- You can query and manage AD DS by using Lightweight Directory Access Protocol (LDAP) calls.
- AD DS primarily uses the Kerberos protocol for authentication.
- AD DS uses OUs and GPOs for management.
- AD DS includes computer objects, representing computers that join an Active Directory domain.
- AD DS uses trusts between domains for delegated management.

You can deploy AD DS on an Azure virtual machine to enable scalability and availability for an on-premises AD DS. However, deploying AD DS on an Azure virtual machine doesn't make any use of Microsoft Entra ID.

Characteristics of Microsoft Entra ID

Although Microsoft Entra ID has many similarities to AD DS, there are also many differences. It's important to realize that using Microsoft Entra isn't the same as deploying an Active Directory domain controller on an Azure virtual machine and adding it to your on-premises domain.

When comparing Microsoft Entra ID with AD DS, it's important to note the following characteristics of Microsoft Entra ID:

- Microsoft Entra ID is primarily an identity solution, and it's designed for internet-based applications by using HTTP (port 80) and HTTPS (port 443) communications.
- Microsoft Entra ID is a multi-tenant directory service.
- Microsoft Entra users and groups are created in a flat structure, and there are no OUs or GPOs.
- You can't query Microsoft Entra ID by using LDAP; instead, Microsoft Entra ID uses the REST API over HTTP and HTTPS.
- Microsoft Entra ID doesn't use Kerberos authentication; instead, it uses HTTP and HTTPS protocols such as SAML, WS-Federation, and OpenID Connect for authentication, and uses OAuth for authorization.
- Microsoft Entra ID includes federation services, and many third-party services such as Facebook are federated with and trust Microsoft Entra ID.

Compare Microsoft Entra ID P1 and P2 plans

The Microsoft Entra ID P1 or P2 tier provides extra functionality as compared to the Free and Office 365 editions. However, premium versions require additional cost per user provisioning. Microsoft Entra ID P1 or P2 comes in two

versions P1 and P2. You can procure it as an extra license or as a part of the Microsoft Enterprise Mobility + Security, which also includes the license for Azure Information Protection and Intune.

Microsoft provides a free trial period that can be used to experience the full functionality of the Microsoft Entra ID P2 edition. The following features are available with the Microsoft Entra ID P1 edition:

- **Self-service group management**. It simplifies the administration of groups where users are given the rights to create and manage the groups. End users can create requests to join other groups, and group owners can approve requests and maintain their groups' memberships.
- **Advanced security reports and alerts**. You can monitor and protect access to your cloud applications by viewing detailed logs that show advanced anomalies and inconsistent access pattern reports. Advanced reports are machine learning based and can help you gain new insights to improve access security and respond to potential threats.
- **Multi-factor authentication**. Full multi-factor authentication (MFA) works with on-premises applications (using virtual private network [VPN], RADIUS, and others), Azure, Microsoft 365, Dynamics 365, and third-party Microsoft Entra gallery applications. It doesn't work with non-browser off-the-shelf apps, such as Microsoft Outlook. Full multi-factor authentication is covered in more detail in the following units in this lesson.
- **Microsoft Identity Manager (MIM) licensing**. MIM integrates with Microsoft Entra ID P1 or P2 to provide hybrid identity solutions. MIM can bridge multiple on-premises authentication stores such as AD DS, LDAP, Oracle, and other applications with Microsoft Entra ID. This provides consistent experiences to on-premises line-of-business (LOB) applications and SaaS solutions.
- **Enterprise SLA of 99.9%**. You're guaranteed at least 99.9% availability of the Microsoft Entra ID P1 or P2 service. The same SLA applies to Microsoft Entra Basic.
- **Password reset with writeback**. Self-service password reset follows the Active Directory on-premises password policy.
- **Cloud App Discovery feature of Microsoft Entra ID**. This feature discovers the most frequently used cloud-based applications.
- **Conditional Access based on device, group, or location**. This lets you configure conditional access for critical resources, based on several criteria.
- **Microsoft Entra Connect Health**. You can use this tool to gain operational insight into Microsoft Entra ID. It works with alerts, performance counters, usage patterns, and configuration settings, and presents the collected information in the Microsoft Entra Connect Health portal.

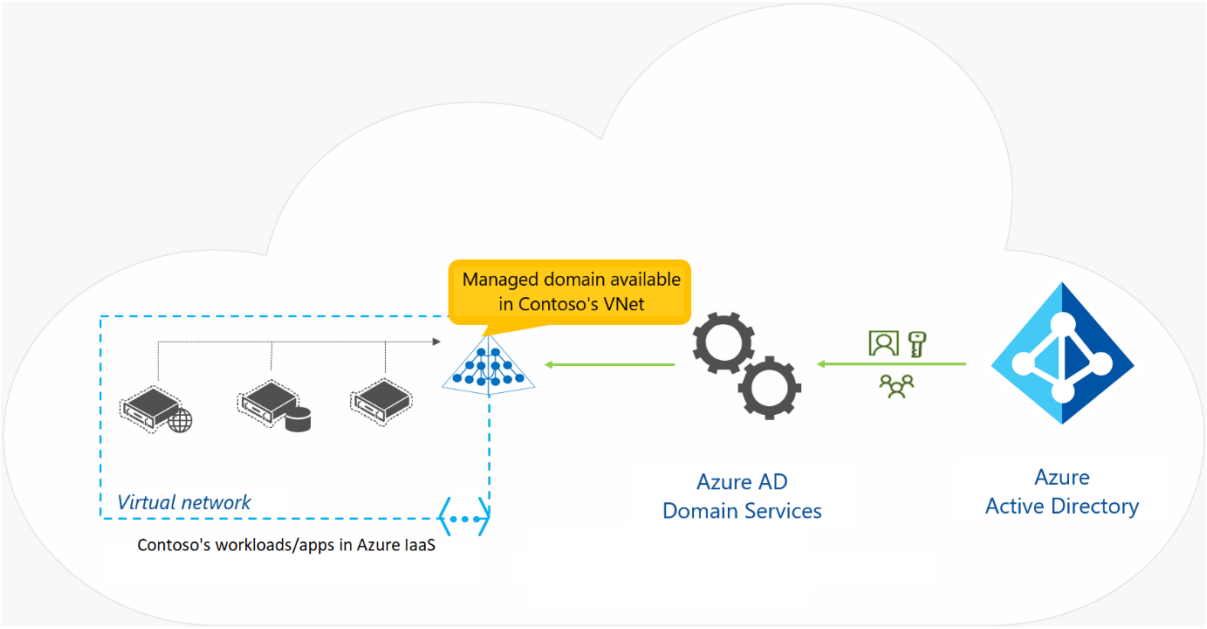
In addition to these features, the Microsoft Entra ID P2 license provides extra functionalities:

- **Microsoft Entra ID Protection**. This feature provides enhanced functionalities for monitoring and protecting user accounts. You can define user risk policies and sign-in policies. In addition, you can review users' behaviour and flag users for risk.
- **Microsoft Entra Privileged Identity Management**. This functionality lets you configure additional security levels for privileged users such as administrators. With Privileged Identity Management, you define permanent and temporary administrators. You also define a policy workflow that activates whenever someone wants to use administrative privileges to perform some tasks.

Examine Microsoft Entra Domain Services

In most organizations today, line-of-business (LOB) applications are deployed on computers and devices that are domain members. These organizations use AD DS–based credentials for authentication, and Group Policy manages them. When you consider moving these apps to run in Azure, one key issue is how to provide authentication services to these apps. To satisfy this need, you can choose to implement a site-to-site virtual private network (VPN) between your local infrastructure and the Azure IaaS, or you can deploy replica domain controllers from your local AD DS as virtual machines (VMs) in Azure. These approaches can entail additional costs and administrative effort. Additionally, the difference between these two approaches is that with the first option, authentication traffic will cross the VPN, while in the second option, replication traffic will cross the VPN and authentication traffic stays in the cloud.

Microsoft provides Microsoft Entra Domain Services as an alternative to these approaches. This service, which runs as part of the Microsoft Entra ID P1 or P2 tier, provides domain services such as Group Policy management, domain joining, and Kerberos authentication to your Microsoft Entra tenant. These services are fully compatible with locally deployed AD DS, so you can use them without deploying and managing additional domain controllers in the cloud.



Because Microsoft Entra ID can integrate with your local AD DS, when you implement Microsoft Entra Connect, users can utilize organizational credentials in both on-premises AD DS and in Microsoft Entra Domain Services. Even if you don't have AD DS deployed locally, you can choose to use Microsoft Entra Domain Services as a cloud-only service. This enables you to have similar functionality of locally deployed AD DS without having to deploy a single domain controller on-premises or in the cloud. For example, an organization can choose to create a Microsoft Entra tenant and enable Microsoft Entra Domain Services and then deploy a virtual network between its on-premises resources and the Microsoft Entra tenant. You can enable Microsoft Entra Domain Services for this virtual network so that all on-premises users and services can use domain services from Microsoft Entra ID.

Microsoft Entra Domain Services provides several benefits for organizations, such as:

- Administrators don't need to manage, update, and monitor domain controllers.
- Administrators don't need to deploy and manage Active Directory replication.
- There's no need to have Domain Admins or Enterprise Admins groups for domains that Microsoft Entra ID manages.

If you choose to implement Microsoft Entra Domain Services, you need to be aware of the service's current limitations. These include:

- Only the base computer Active Directory object is supported.

- It’s not possible to extend the schema for the Microsoft Entra Domain Services domain.
- The organizational unit (OU) structure is flat and nested OUs aren't currently supported.
- There’s a built-in Group Policy Object (GPO), and it exists for computer and user accounts.
- It’s not possible to target OUs with built-in GPOs. Additionally, you can't use Windows Management Instrumentation filters or security-group filtering.

By using Microsoft Entra Domain Services, you can freely migrate applications that use LDAP, NTLM, or the Kerberos protocols from your on-premises infrastructure to the cloud. You can also use applications such as Microsoft SQL Server or Microsoft SharePoint Server on VMs or deploy them in the Azure IaaS, without needing domain controllers in the cloud or a VPN to local infrastructure.

You can enable Microsoft Entra Domain Services by using the Azure portal. This service charges per hour based on the size of your directory.

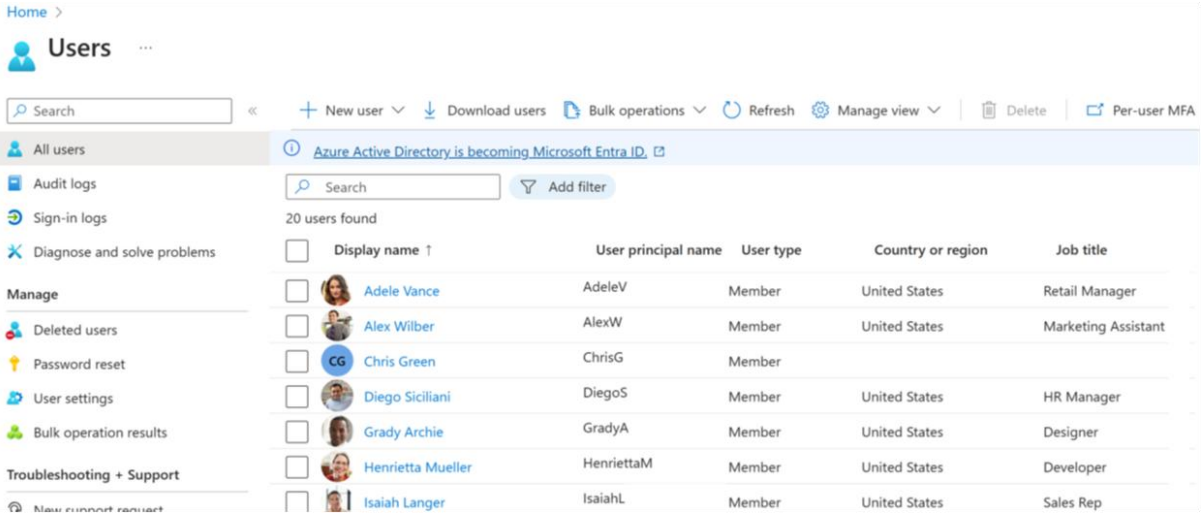
Create, configure, and manage users

Every user who needs access to Azure resources needs an Azure user account. A user account contains all the information needed to authenticate the user during the sign-on process. Once authenticated Microsoft Entra ID builds an access token to authorize the user and determine what resources they can access and what they can do with those resources.

You use the **Microsoft Entra ID** dashboard in the Azure portal to work with user objects. Keep in mind that you can only work with a single directory at a time. You can use the **Directory + Subscription** panel to switch directories. The dashboard also has a **Switch directory** button in the toolbar which makes it easy to switch to another available directory.

View users

To view the Microsoft Entra users, select the **Users** entry under **Identity** - then open the **All-Users** view. Take a minute to access the portal and view your users. Notice the **User Type** column to see members and guests, as the following figure depicts.



Typically, Microsoft Entra ID defines users in three ways:

- **Cloud identities** - These users exist only in Microsoft Entra ID. Examples are administrator accounts and users that you manage yourself. Their source is **Microsoft Entra ID** or **External Microsoft Entra directory** if the user is defined in another Microsoft Entra instance but needs access to subscription resources controlled by this directory. When these accounts are removed from the primary directory, they're deleted.

- **Directory-synchronized identities** - These users exist in an on-premises Active Directory. A synchronization activity that occurs via **Microsoft Entra Connect** brings these users in to Azure. Their source is **Windows Server AD**.
- **Guest users** - These users exist outside Azure. Examples are accounts from other cloud providers and Microsoft accounts such as an Xbox LIVE account. Their source is **Invited user**. This type of account is useful when external vendors or contractors need access to your Azure resources. Once their help is no longer necessary, you can remove the account and all of their access.

Create, configure, and manage groups

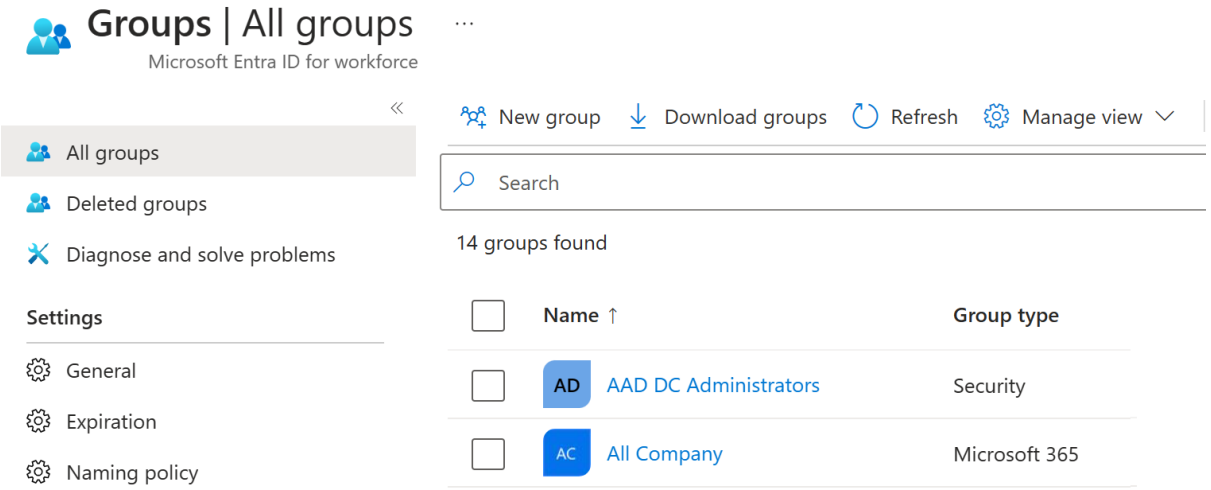
A Microsoft Entra group helps organize users, which makes it easier to manage permissions. Using groups lets the resource owner (or Microsoft Entra directory owner), assign a set of access permissions to all the members of the group, instead of having to provide the rights one-by-one. Groups allow us to define a security boundary and then add and remove specific users to grant or deny access with a minimum amount of effort. Even better, Microsoft Entra ID supports the ability to define membership based on rules - such as what department a user works in, or the job title they have.

Microsoft Entra ID allows you to define two different types of groups.

- **Security groups** - the most common type of groups and are used to manage member and computer access to shared resources for a group of users. For example, you can create a security group for a specific security policy. By doing it this way, you can give a set of permissions to all the members at once, instead of having to add permissions to each member individually. This option requires a Microsoft Entra administrator.
- **Microsoft 365 groups** - provide collaboration opportunities by giving members access to a shared mailbox, calendar, files, SharePoint site, and more. This option also lets you give people outside of your organization access to the group. This option is available to users as well as admins.

View available groups

You can view all groups through the **Groups** item under **Identity** in the Microsoft Entra admin center. A new Microsoft Entra ID deployment won't have any groups defined.

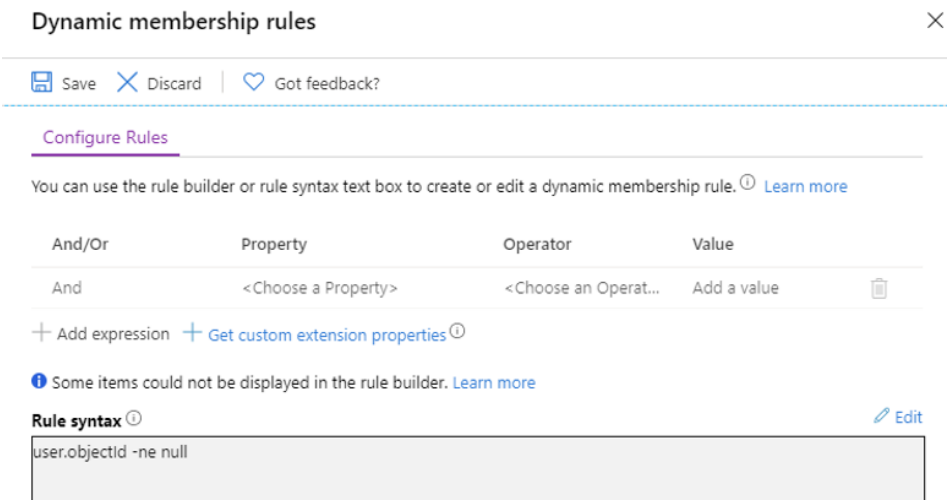


The second characteristic of a group that you need to be aware of is the **Membership Type**. This specifies how individuals members are added to the group. The two types are:

- Assigned - members are added and maintained manually.
- Dynamic - members are added based on rules, creating a Dynamic Group. These groups are still either a security group or Microsoft 365 group, just their members are controlled by rule.

Dynamic groups

The final type of group is a dynamic group, which the name implies, the membership is generated by a formula each time the group is used. A dynamic group includes any recipient in Active Directory with attribute values that match its filter. If a recipient's properties are modified to match the filter, the recipient could inadvertently become a group member and start receiving messages that are sent to the group. Well-defined, consistent account provisioning processes will reduce the chances of this issue occurring.



This dynamic group would consist of all valid members of the Microsoft Entra ID.

What is Microsoft Azure

Azure is a continually expanding set of cloud services that help you meet current and future business challenges. Azure gives you the freedom to build, manage, and deploy applications on a massive global network using your favorite tools and frameworks.

What does Azure offer?

Limitless innovation. Build intelligent apps and solutions with advanced technology, tools, and services to take your business to the next level. Seamlessly unify your technology to simplify platform management and to deliver innovations efficiently and securely on a trusted cloud.

- **Bring ideas to life:** Build on a trusted platform to advance your organization with industry-leading AI and cloud services.
- **Seamlessly unify:** Efficiently manage all your infrastructure, data, analytics, and AI solutions across an integrated platform.
- **Innovate on trust:** Rely on trusted technology from a partner who's dedicated to security and responsibility.

What can I do with Azure?

Azure provides more than 100 services that enable you to do everything from running your existing applications on virtual machines to exploring new software paradigms, such as intelligent bots and mixed reality.

Many teams start exploring the cloud by moving their existing applications to virtual machines (VMs) that run in Azure. Migrating your existing apps to VMs is a good start, but the cloud is much more than a different place to run your VMs.

For example, Azure provides artificial intelligence (AI) and machine-learning (ML) services that can naturally communicate with your users through vision, hearing, and speech. It also provides storage solutions that dynamically grow to accommodate massive amounts of data. Azure services enable solutions that aren't feasible without the power of the cloud.

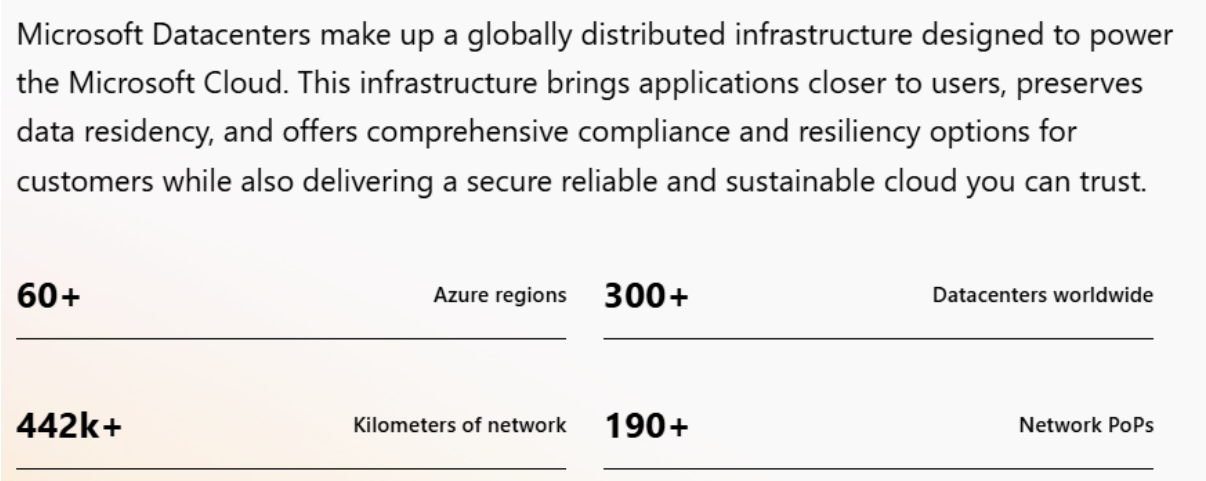
Describe Azure physical infrastructure

The core architectural components of Azure may be broken down into two main groupings: the physical infrastructure, and the management infrastructure.

Physical infrastructure

The physical infrastructure for Azure starts with datacenters. Conceptually, the datacenters are the same as large corporate datacenters. They're facilities with resources arranged in racks, with dedicated power, cooling, and networking infrastructure.

As a global cloud provider, Azure has datacenters around the world. However, these individual datacenters aren't directly accessible. Datacenters are grouped into Azure Regions or Azure Availability Zones that are designed to help you achieve resiliency and reliability for your business-critical workloads.



Regions

A region is a geographical area on the planet that contains at least one, but potentially multiple datacenters that are nearby and networked together with a low-latency network. Azure intelligently assigns and controls the resources within each region to ensure workloads are appropriately balanced.

Note

Some services or virtual machine (VM) features are only available in certain regions, such as specific VM sizes or storage types. There are also some global Azure services that don't require you to select a particular region, such as Microsoft Entra ID, Azure Traffic Manager, and Azure DNS.

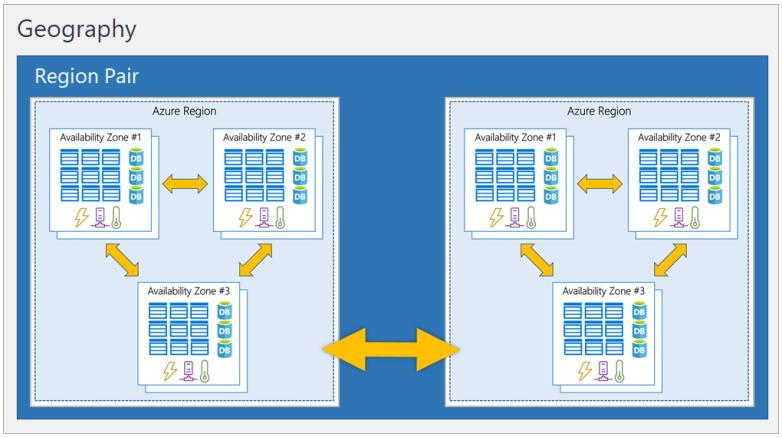
Region pairs

Most Azure regions are paired with another region within the same geography (such as US, Europe, or Asia) at least 300 miles away. This approach allows for the replication of resources across a geography that helps reduce the likelihood of interruptions because of events such as natural disasters, civil unrest, power outages, or physical network outages that affect an entire region. For example, if a region in a pair was affected by a natural disaster, services would automatically fail over to the other region in its region pair.

Important

Not all Azure services automatically replicate data or automatically fall back from a failed region to cross-replicate to another enabled region. In these scenarios, recovery and replication must be configured by the customer.

Examples of region pairs in Azure are West US paired with East US and South-East Asia paired with East Asia. Because the pair of regions are directly connected and far enough apart to be isolated from regional disasters, you can use them to provide reliable services and data redundancy.



Additional advantages of region pairs:

- If an extensive Azure outage occurs, one region out of every pair is prioritized to make sure at least one is restored as quickly as possible for applications hosted in that region pair.
- Planned Azure updates are rolled out to paired regions one region at a time to minimize downtime and risk of application outage.
- Data continues to reside within the same geography as its pair (except for Brazil South) for tax- and law-enforcement jurisdiction purposes.

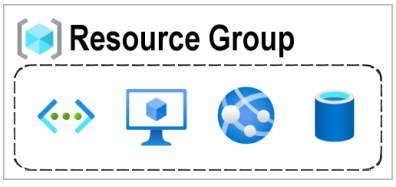
Describe Azure management infrastructure

The management infrastructure includes Azure resources and resource groups, subscriptions, and accounts. Understanding the hierarchical organization will help you plan your projects and products within Azure.

Azure resources and resource groups

A resource is the basic building block of Azure. Anything you create, provision, deploy, etc. is a resource. Virtual Machines (VMs), virtual networks, databases, cognitive services, etc. are all considered resources within Azure.

Resource groups are simply groupings of resources. When you create a resource, you're required to place it into a resource group. While a resource group can contain many resources, a single resource can only be in one resource group at a time. Some resources may be moved between resource groups, but when you move a resource to a new group, it will no longer be associated with the former group. Additionally, resource groups can't be nested, meaning you can't put resource group B inside of resource group A.



Resource groups provide a convenient way to group resources together. When you apply an action to a resource group, that action will apply to all the resources within the resource group. If you delete a resource group, all the resources will be deleted. If you grant or deny access to a resource group, you've granted or denied access to all the resources within the resource group.

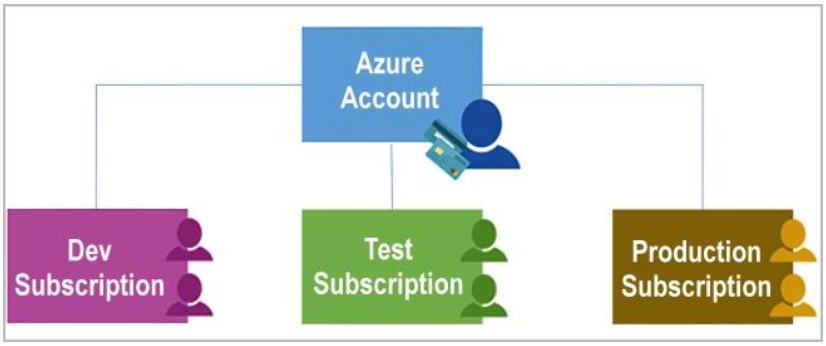
When you're provisioning resources, it's good to think about the resource group structure that best suits your needs.

For example, if you're setting up a temporary dev environment, grouping all the resources together means you can deprovision all of the associated resources at once by deleting the resource group. If you're provisioning compute resources that will need three different access schemas, it may be best to group resources based on the access schema, and then assign access at the resource group level.

There aren't hard rules about how you use resource groups, so consider how to set up your resource groups to maximize their usefulness for you.

Azure subscriptions

In Azure, subscriptions are a unit of management, billing, and scale. Similar to how resource groups are a way to logically organize resources, subscriptions allow you to logically organize your resource groups and facilitate billing.



Using Azure requires an Azure subscription. A subscription provides you with authenticated and authorized access to Azure products and services. It also allows you to provision resources. An Azure subscription links to an Azure account, which is an identity in Microsoft Entra ID or in a directory that Microsoft Entra ID trusts.

An account can have multiple subscriptions, but it's only required to have one. In a multi-subscription account, you can use the subscriptions to configure different billing models and apply different access-management policies. You can use Azure subscriptions to define boundaries around Azure products, services, and resources. There are two types of subscription boundaries that you can use:

- **Billing boundary**: This subscription type determines how an Azure account is billed for using Azure. You can create multiple subscriptions for different types of billing requirements. Azure generates separate billing reports and invoices for each subscription so that you can organize and manage costs.
- **Access control boundary**: Azure applies access-management policies at the subscription level, and you can create separate subscriptions to reflect different organizational structures. An example is that within a business, you have different departments to which you apply distinct Azure subscription policies. This billing model allows you to manage and control access to the resources that users provision with specific subscriptions.

Create additional Azure subscriptions

Similar to using resource groups to separate resources by function or access, you might want to create additional subscriptions for resource or billing management purposes. For example, you might choose to create additional subscriptions to separate:

- **Environments**: You can choose to create subscriptions to set up separate environments for development and testing, security, or to isolate data for compliance reasons. This design is particularly useful because resource access control occurs at the subscription level.
- **Organizational structures**: You can create subscriptions to reflect different organizational structures. For example, you could limit one team to lower-cost resources, while allowing the IT department a full range. This design allows you to manage and control access to the resources that users provision within each subscription.
- **Billing**: You can create additional subscriptions for billing purposes. Because costs are first aggregated at the subscription level, you might want to create subscriptions to manage and track costs based on your needs. For instance, you might want to create one subscription for your production workloads and another subscription for your development and testing workloads.

Azure management groups

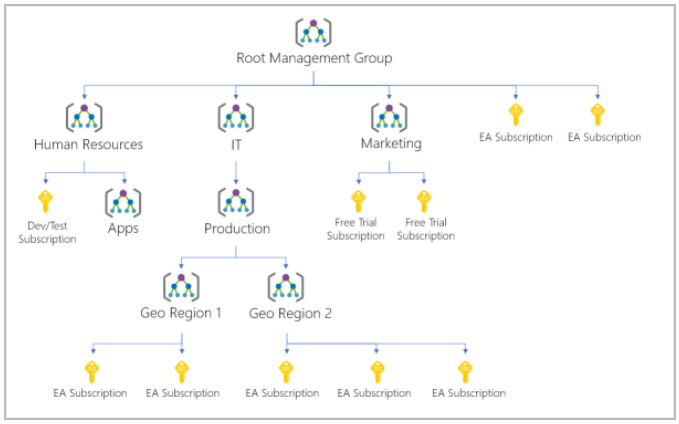
The final piece is the management group. Resources are gathered into resource groups, and resource groups are gathered into subscriptions. If you're just starting in Azure that might seem like enough hierarchy to keep things

organized. But imagine if you're dealing with multiple applications, multiple development teams, in multiple geographies.

If you have many subscriptions, you might need a way to efficiently manage access, policies, and compliance for those subscriptions. Azure management groups provide a level of scope above subscriptions. You organize subscriptions into containers called management groups and apply governance conditions to the management groups. All subscriptions within a management group automatically inherit the conditions applied to the management group, the same way that resource groups inherit settings from subscriptions and resources inherit from resource groups. Management groups give you enterprise-grade management at a large scale, no matter what type of subscriptions you might have. Management groups can be nested.

Management group, subscriptions, and resource group hierarchy

You can build a flexible structure of management groups and subscriptions to organize your resources into a hierarchy for unified policy and access management. The following diagram shows an example of creating a hierarchy for governance by using management groups.



Some examples of how you could use management groups might be:

- **Create a hierarchy that applies a policy**. You could limit VM locations to the US West Region in a group called Production. This policy will inherit onto all the subscriptions that are descendants of that management group and will apply to all VMs under those subscriptions. This security policy can't be altered by the resource or subscription owner, which allows for improved governance.
- **Provide user access to multiple subscriptions**. By moving multiple subscriptions under a management group, you can create one Azure role-based access control (Azure RBAC) assignment on the management group. Assigning Azure RBAC at the management group level means that all sub-management groups, subscriptions, resource groups, and resources underneath that management group would also inherit those permissions. One assignment on the management group can enable users to have access to everything they need instead of scripting Azure RBAC over different subscriptions.

Important facts about management groups:

- 10,000 management groups can be supported in a single directory.
- A management group tree can support up to six levels of depth. This limit doesn't include the root level or the subscription level.
- Each management group and subscription can support only one parent.

Azure Policy initiatives

Azure Policy is a service that allows you to create, assign, and manage governance policies that enforce rules and effects over Azure resources to ensure that they stay compliant with your IT governance standards. These policies enforce various rules and effects on resources, ensuring that they adhere to corporate standards and service-level agreements. These policies are described in JSON format and are known as policy definitions. Azure Policy is crucial for enforcing organizational standards and assessing compliance on a large scale.

Azure Policy initiatives are a collection of Azure Policy definitions that are grouped together toward a specific goal or purpose. By consolidating multiple Azure policies into a single item, Azure Policy initiatives allow centralized control and enforcement of configurations across Azure resources.

Industry-specific organizations in sectors like government, public sector, finance, and others accelerate digital transformation and achieve better business outcomes. They can achieve this objective by adopting specific, sovereignty-focused Azure Policy initiatives to address the complexity of compliance with national and regional regulatory requirements.

Customers can create Azure Policy initiatives that aid in customizing deployments to reduce the time needed to audit environments and help meet established regulatory compliance frameworks and government requirements. The initiatives help public sector partners and customers create cloud guardrails and enforce specific regulations effectively. They can layer policy initiatives to help form a complete solution for their specific needs, and they can use deployment automation to ensure consistency, best practices, and save time.

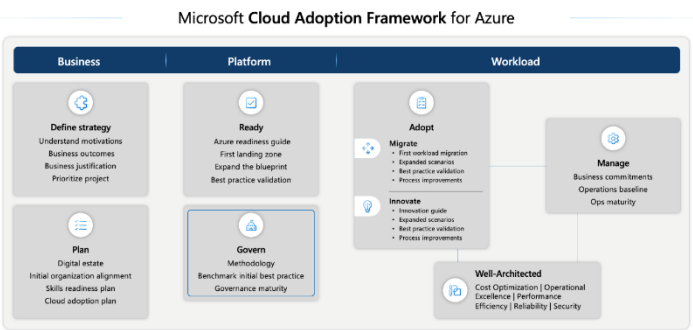
In this module, you learn how Azure Policy can help do the common tasks related to creating, assigning, and managing policies across your organization, such as:

- Assign a policy to enforce a condition for resources you create in the future
- Create and assign an initiative definition to track compliance for multiple resources
- Resolve a noncompliant or denied resource
- Implement a new policy across an organization

Cloud Adoption Framework for Azure

The Microsoft Cloud Adoption Framework for Azure offers comprehensive technical guidance for Microsoft Azure. This end-to-end framework helps cloud architects, IT experts, and business leaders reach their cloud adoption objectives. Cloud Adoption Framework includes best practices, documentation, and tools that Microsoft employees, partners, and customers contribute to formulate and implement effective business and technology strategies for the cloud. For more information, see Microsoft Cloud Adoption Framework for Azure documentation - Cloud Adoption Framework | Microsoft Learn.

The following diagram provides high-level information about different methodologies that are included in Microsoft Cloud Adoption Framework for Azure for each phase of your cloud adoption life cycle. Within the framework, Microsoft Azure Policy plays a significant role in the governance framework to help govern your cloud environment and workloads.



Cloud governance refers to the management of cloud usage in your organization. The Cloud Adoption Framework - Govern methodology offers a systematic framework for setting up and improving cloud governance in Azure. This guidance applies to organizations across various industries and it addresses crucial areas, such as regulatory compliance, security, operations, cost management, data, resource management, and AI. It's essential for defining and maintaining efficient cloud use.

Comprehensive cloud governance oversees all aspects of cloud use, minimizes various risks (such as compliance, security, resource management, and data-related concerns), and optimizes cloud operations throughout the organization. It ensures that cloud activities are consistent with the overall cloud strategy, facilitating the achievement of business objectives with reduced obstacles.

Steps for cloud governance

Cloud governance is a continuous process. It requires ongoing monitoring, evaluation, and adjustments to adapt to evolving technologies, risks, and compliance requirements. The Cloud Adoption Framework - Govern methodology divides cloud governance into five steps.

1. **Build a governance team** - Establish a dedicated cloud governance team that's responsible for defining, maintaining, and reporting on the progress of cloud governance policies.
2. **Assess cloud risks** - Conduct a thorough risk assessment that's unique to your organization, addressing all risk categories, including regulatory compliance, security, operations, costs, data management, resource management, and AI-related risks.
3. **Document cloud governance policies** - Clearly document cloud governance policies that dictate acceptable cloud usage and outline the rules and guidelines that mitigate identified risks.
4. **Enforce cloud governance policies** - Implement a systematic approach to ensure compliance with cloud governance policies. Use automated tools alongside manual oversight to enforce compliance. These tools help set guardrails, monitor configurations, and ensure adherence to policies.
5. **Monitor cloud governance** - Regularly monitor cloud usage and the governance teams to ensure ongoing compliance with the established cloud governance policies.

You should complete all five steps to establish cloud governance and regularly iterate on steps 2-5 to maintain cloud governance over time.

Considerations for defining a cloud governance policy

The key considerations when defining a corporate cloud governance policy are as follows:

- **Business risk** – You must document the evolving business risks and the business's tolerance for risk based on data classification and application criticality.
- **Policy and compliance** – You must convert risk decisions into policy statements to establish cloud adoption boundaries efficiently.
- **Process** – You must establish processes to monitor violations and adherence to corporate policies.

The five core disciplines of cloud governance are as follows:

- **Cost management** – Evaluates and monitors costs, including controlling IT expenditures to establish well-defined cost management. It also includes adjusting resources according to demand. It's crucial to exercise control over cloud expenditure to derive greater value from your investments.
- **Security baseline** – Ensures compliance with IT security requirements by applying a security baseline to all adoption efforts.
- **Resource consistency** – Ensures consistency in resource configuration and enforcing practices for onboarding, recovery, and discoverability.
- **Identity baseline** – Ensures that the baseline for identity and access is enforced by consistently applying role definitions and assignments.
- **Deployment acceleration** – Accelerates the deployment of policies through centralization, consistency, and standardization across deployment templates.

Adopting these measures simplifies the compliance process and also enhances the security and efficiency of your cloud environment.

Cloud governance with Azure Policy

Azure's primary governance tool is Azure Policy. Azure Policy facilitates the governance of all resources, including current and forthcoming resources. It helps to enforce organizational standards and to assess compliance at scale by establishing guardrails across various resources.

Azure Policy allows for centralized management, allowing you to track compliance status and investigate changes leading to noncompliance. You can consolidate all compliance data into a singular repository, which simplifies auditing processes for effective cloud compliance and resource governance. The compliance dashboard offered by Azure Policy presents an aggregated view of the environment's overall state, with the ability to examine details at each resource and each policy level.

Azure Policy ensures consistent adherence to compliance standards and prevents misconfigurations. It also helps bring your resources to compliance through bulk remediation for existing resources and automatic remediation for new resources. Additionally, embedding Azure Policy directly into the Azure platform can significantly reduce the need for external approval processes, thus boosting developer productivity.

Some useful governance actions that you can enforce with Azure Policy include:

- Ensure that your team deploys Azure resources only to allowed regions.
- Enforce geo-replication rules to comply with your data residency requirements.
- Allow only certain virtual machine sizes for your cloud environment.
- Enforce the consistent application of taxonomic tags across resources.
- Recommend system updates on your servers.
- Allow multifactor authentication for all subscription accounts.
- Require resources to send diagnostic logs to an Azure Monitor Logs workspace.

Azure Policy evaluates your resources and highlights resources that aren't compliant with the policies that you create. Azure Policy can also prevent noncompliant resources from being created. Azure Policy comes with built-in policy and initiative definitions for Storage, Networking, Compute, Security Center, and Monitoring. For example, if you define a policy that only allows a certain size for the virtual machines (VMs) to be used in your environment, that policy is

invoked when you create a new VM and whenever you resize existing VMs. Azure Policy also evaluates and monitors all current VMs in your environment, including VMs that were created before the policy was created.

In some cases, Azure Policy can automatically remediate noncompliant resources and configurations to ensure the integrity of the state of the resources. For example, if all resources in a specific resource group need to have the *AppName* tag with a value of *SpecialOrders*, Azure Policy can automatically apply that tag if it's missing. However, you maintain full control over your environment. If there's a particular resource that you don't want Azure Policy to automatically update, you can mark that resource as an exception, and the policy won't automatically update it.

Azure Policy also integrates with Azure DevOps by applying any continuous integration and delivery pipeline policies that pertain to the pre-deployment and post-deployment phases of your applications.

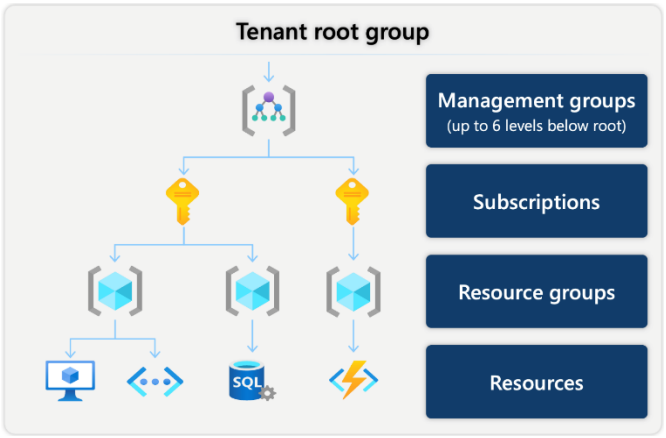
The objective when designing an Azure Policy should be to strike a balance between control and stability in one area with speed and results in the other. The main reason to maintain balance is to ensure that the environment remains highly manageable so that you can implement necessary levels of control to ensure governance without hindering operational efficiency or productivity. In establishing and enforcing these controls, you must take care that your speed to achieve efficiency isn't affected. Hence, achieving this balance between control and stability with speed and results, often requires thoughtful compromise, and it's necessary to carefully evaluate the potential impact before introducing new policies.

Azure Policy design principles

Governance provides mechanisms and processes to maintain control over your applications and resources in Azure. It involves planning your policy in Azure Policy and setting strategic priorities. While designing your policy, you must organize your cloud-based resources to secure, manage, and track costs that are related to your workloads.

Hierarchy for governance

Azure provides four levels of management to establish proper governance: Management groups, Subscriptions, Resource groups, and Resources. You can build a flexible structure of management groups and subscriptions to organize your resources into a hierarchy for unified policy and access management. The following diagram shows an example of creating a hierarchy for governance by using management groups.



The Azure structure starts with the tenant root group at the top, followed by a hierarchy of management groups, which can extend to six levels beneath the root. The definition of each level in the management hierarchy and relationship between these levels is as follows:

Introduction to Azure Resource Manager

Azure Resource Manager is the deployment and management service for Azure. It provides a management layer that allows you to create, update, and delete resources in your Azure account.

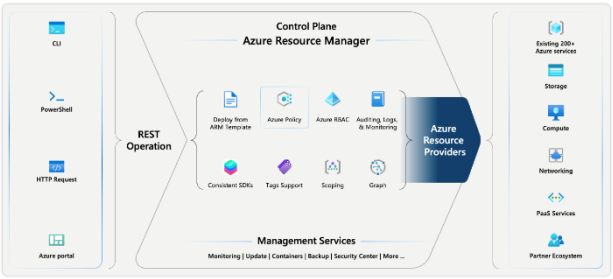
Azure operations are classified into two main types: control plane and data plane. The control plane helps you manage resources in your subscription, while the data plane allows you to access the capabilities provided by instances of specific resource types.

Control plane

Azure Policy operates in the control plane to enforce rules and compliance on your resources. Azure Resource Manager manages all control plane operations in Azure and includes the different components that are centralized between the different services. Azure Policy is integrated with Azure Resource Manager.

Azure Resource Manager manages essential functions, such as template-based deployments, role-based access control (RBAC), auditing, monitoring, and tagging, which provides a unified management experience for Azure resources after deployment. For example, consider a scenario where you have a storage account. With Azure Resource Manager, you can create the storage account and enforce a policy that mandates encryption for all storage accounts.

Azure Policy and Azure Resource Manager



When you send a control plane request through any of the Azure APIs, tools, or SDKs, Azure Resource Manager receives the request. All capabilities that are available in the portal are also available through PowerShell, Azure CLI, REST APIs, and client SDKs. These tools use the same API to process requests, ensuring uniform results and capabilities. The Resource Manager authenticates and authorizes the request before forwarding it to the appropriate Azure resource provider that completes the operation.

What is Azure RBAC?

When it comes to identity and access, most organizations that are considering using the public cloud are concerned about two things:

1. Ensuring that when people leave the organization, they lose access to resources in the cloud.
2. Striking the right balance between autonomy and central governance. For example, giving project teams the ability to create and manage virtual machines in the cloud, while centrally controlling the networks those VMs use to communicate with other resources.

Microsoft Entra ID and Azure RBAC work together to make it simple to carry out these goals.

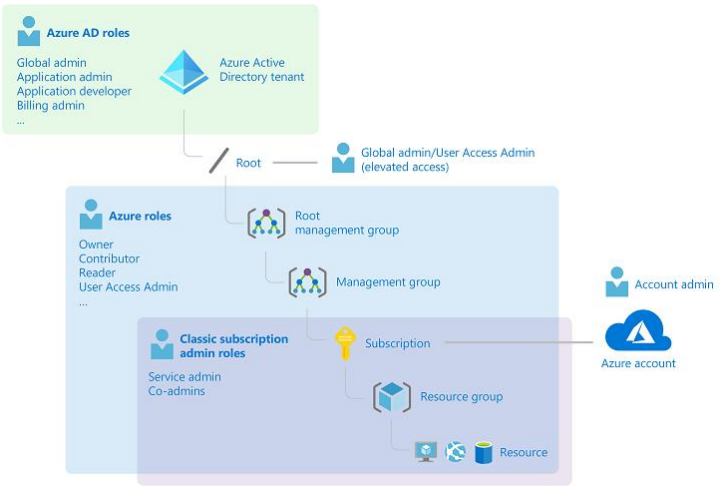
Azure subscriptions

First, remember that each Azure subscription is associated with a single Microsoft Entra directory. Users, groups, and applications in that directory can manage resources in the Azure subscription. The subscriptions use Microsoft Entra ID for single sign-on (SSO) and access management. You can extend your on-premises Active Directory to the cloud by using **Microsoft Entra Connect**. This feature allows your employees to manage their Azure subscriptions by using their existing work identities. When you disable an on-premises Active Directory account, it automatically loses access to all Azure subscriptions connected with Microsoft Entra ID.

What's Azure RBAC?

Azure RBAC is an authorization system built on Azure Resource Manager that provides fine-grained access management for resources in Azure. With Azure RBAC, you can grant the exact access that users need to do their jobs. For example, you can use Azure RBAC to let one employee manage virtual machines in a subscription, while another manages SQL databases within the same subscription.

The following diagram depicts how the classic subscription administrator roles, Azure roles, and Microsoft Entra roles are related at a high level. Child scopes, such as service instances, inherit roles assigned at a higher scope, like an entire subscription.



In the preceding diagram, a subscription is associated with only one Microsoft Entra tenant. Also note that a resource group can have multiple resources, but it's associated with only one subscription. Although it's not obvious from the diagram, a resource can be bound to only one resource group.

What can I do with Azure RBAC?

Azure RBAC allows you to grant access to Azure resources that you control. Suppose you need to manage access to resources in Azure for the development, engineering, and marketing teams. You've started to receive access requests, and you need to quickly learn how access management works for Azure resources.

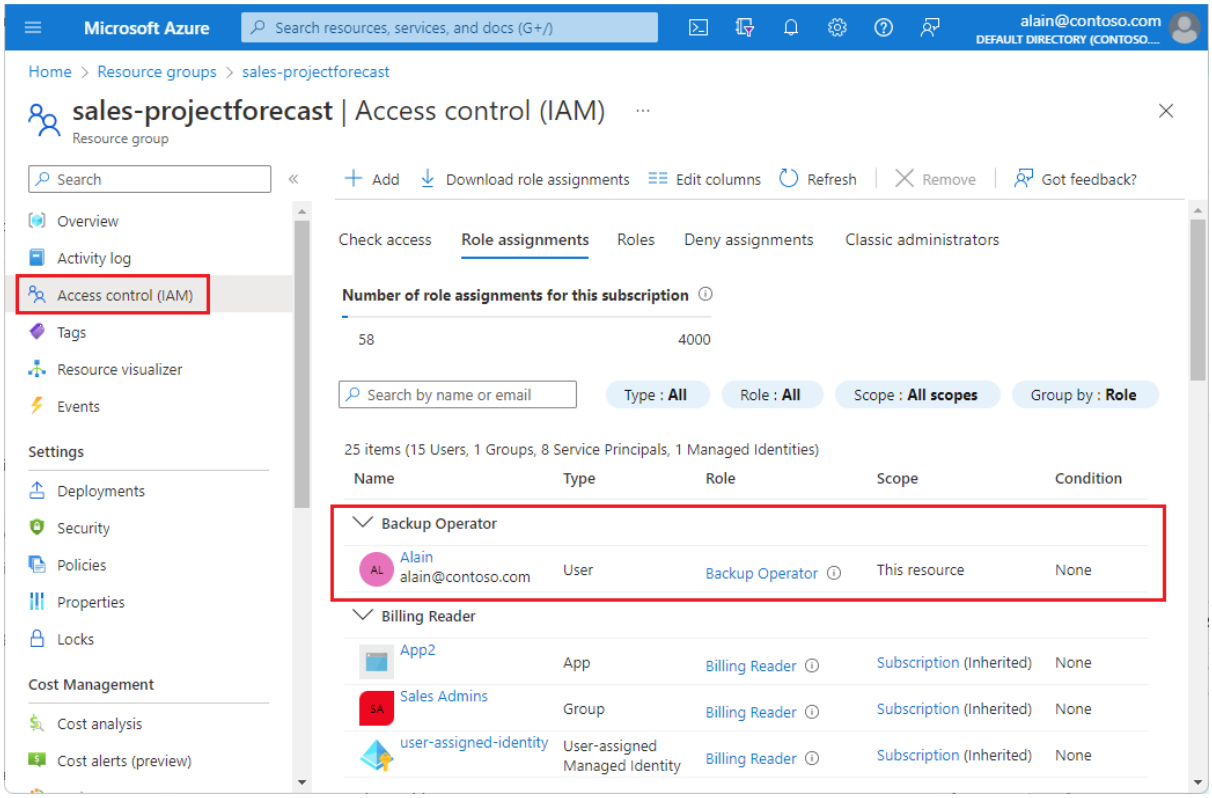
Here are some scenarios you can implement with Azure RBAC:

- Allow one user to manage virtual machines in a subscription and another user to manage virtual networks.
- Allow a database administrator group to manage SQL databases in a subscription.
- Allow a user to manage all resources in a resource group, such as virtual machines, websites, and subnets.
- Allow an application to access all resources in a resource group.

Azure RBAC in the Azure portal

In several areas in the Azure portal, you'll see a pane named **Access control (IAM)**, also known as *identity and access management*. On this pane, you can see who has access to that area and their role. Using this same pane, you can grant or remove access.

The following shows an example of the Access control (IAM) pane for a resource group. In this example, Alain has been assigned the Backup Operator role for this resource group.

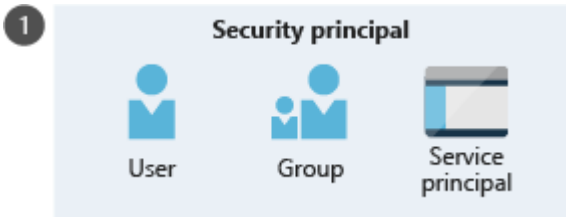


How does Azure RBAC work?

You can control access to resources using Azure RBAC by creating role assignments, which control how permissions are enforced. To create a role assignment, you need three elements: a security principal, a role definition, and a scope. You can think of these elements as *who*, *what*, and *where*.

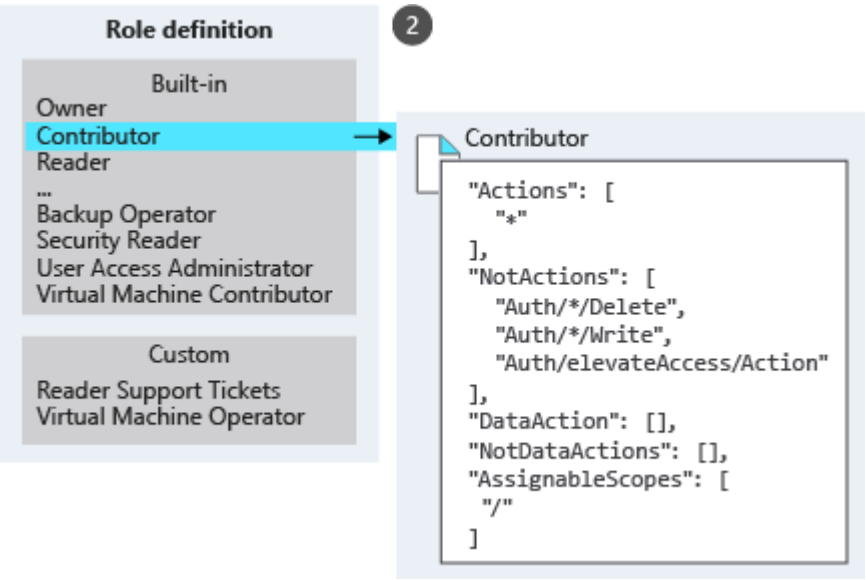
1. Security principal (who)

A *security principal* is just a fancy name for a user, group, or application to which you want to grant access.



2. Role definition (what)

A *role definition* is a collection of permissions. It's sometimes just called a role. A role definition lists the permissions the role can perform such as read, write, and delete. Roles can be high-level, like Owner, or specific, like Virtual Machine Contributor.



Azure includes several built-in roles that you can use. The following lists four fundamental built-in roles:

- **Owner**: Has full access to all resources, including the right to delegate access to others.
- **Contributor**: Can create and manage all types of Azure resources, but can't grant access to others.
- **Reader**: Can view existing Azure resources.
- **User Access Administrator**: Lets you manage user access to Azure resources.

If the built-in roles don't meet the specific needs of your organization, you can create your own custom roles.

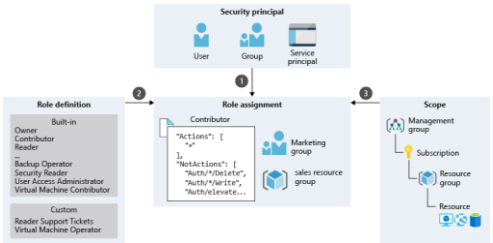
3. Scope (where)

Scope is the level where the access applies. This is helpful if you want to make someone a Website Contributor but only for one resource group.

In Azure, you can specify a scope at multiple levels: management group, subscription, resource group, or resource. Scopes are structured in a parent-child relationship. When you grant access at a parent scope, the child scopes automatically inherit those permissions. For example, if a group is assigned the Contributor role at the subscription scope, it will inherit the role for all resource groups and resources within the subscription.

Role assignment

Once you have determined the who, what, and where, you can combine those elements to grant access. A *role assignment* is the process of binding a role to a security principal at a particular scope for the purpose of granting access. To grant access, you'll create a role assignment. To revoke access, you'll remove a role assignment.



The following example shows how the Marketing group has been assigned the Contributor role at the sales resource group scope.

Azure RBAC is an allow model

Azure RBAC is an *allow* model. This means that when you're assigned a role, Azure RBAC allows you to perform certain actions such as read, write, or delete. If one role assignment grants you read permissions to a resource group, and a

different role assignment grants you write permissions to the same resource group, then you'll have read and write permissions on that resource group.

Azure RBAC has something called NotActions permissions. You can use NotActions to create a set of not allowed permissions. The access a role grants—the *effective permissions*—is computed by subtracting the NotActions operations from the Actions operations. For example, the Contributor role has both Actions and NotActions. The wildcard (*) in Actions indicates that it can perform all operations on the control plane. You'd then subtract the following operations in NotActions to compute the effective permissions:

- Delete roles and role assignments
- Create roles and role assignments
- Grant the caller User Access Administrator access at the tenant scope
- Create or update any blueprint artifacts
- Delete any blueprint artifacts

What is self-service password reset in Microsoft Entra ID?

You've been asked to assess ways to reduce help-desk costs in your retail organization. You've noticed that the support staff spends a lot of their time resetting passwords for users. Users often complain about delays with this process, and these delays impact their productivity. You want to understand how you can configure Azure to allow users to manage their own passwords.

In this unit, you'll learn how self-service password reset (SSPR) works in Microsoft Entra ID.

Why use SSPR?

In Microsoft Entra ID, any user can change their password if they're already signed in. But if they're not signed in, forgot their password, or it's expired, they'll need to reset their password. With SSPR, users can reset their passwords in a web browser or from a Windows sign-in screen to regain access to Azure, Microsoft 365, and any other application that uses Microsoft Entra ID for authentication.

SSPR reduces the load on administrators because users can fix password problems themselves without having to call the help desk. Also, it minimizes the productivity impact of a forgotten or expired password. Users don't have to wait until an administrator is available to reset their password.

How SSPR works

The user initiates a password reset either by going directly to the password-reset portal, or by selecting the **Can't access your account** link on a sign-in page. The reset portal takes these steps:

1. **Localization**: The portal checks the browser's locale setting and renders the SSPR page in the appropriate language.
2. **Verification**: The user enters their username and passes a CAPTCHA to ensure that it's a user and not a bot.
3. **Authentication**: The user enters the required data to authenticate their identity. They might enter a code or answer security questions.
4. **Password reset**: If the user passes the authentication tests, they can enter a new password and confirm it.
5. **Notification**: A message is sent to the user to confirm the reset.

There are several ways you can customize the SSPR user experience. For example, you can add your company logo to the sign-in page so users know they're in the right place to reset their password.

Authenticate a password reset

It's critical to verify a user's identity before you allow a password reset. Malicious users might exploit any weakness in the system to impersonate that user. Azure supports six different ways to authenticate reset requests.

License requirements

There are two editions of Microsoft Entra ID, Premium P1 and Premium P2. The password-reset functionality you can use depends on your edition.

Any user who is signed in can change their password, regardless of the edition of Microsoft Entra ID.

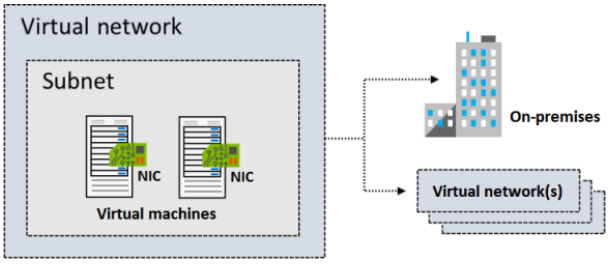
What if you're not signed in, and you've forgotten your password or your password has expired? In this case, you can use SSPR in Microsoft Entra ID P1 or P2. It's also available with Microsoft 365 Apps for business or Microsoft 365.

In a hybrid situation, where you have Active Directory on-premises and Microsoft Entra ID in the cloud, any password change in the cloud must be written back to the on-premises directory. This writeback support is available in Microsoft Entra ID P1 or P2. It's also available with Microsoft 365 Apps for business.

Azure virtual networks

You can implement Azure Virtual Network to create a virtual representation of your network in the cloud. Let's examine some characteristics of virtual networks in Azure.

- An Azure virtual network is a logical isolation of the Azure cloud resources.
- You can use virtual networks to provision and manage virtual private networks (VPNs) in Azure.
- Each virtual network has its own Classless Inter-Domain Routing (CIDR) block and can be linked to other virtual networks and on-premises networks.



- You can link virtual networks with an on-premises IT infrastructure to create hybrid or cross-premises solutions, when the CIDR blocks of the connecting networks don't overlap.
- You control the DNS server settings for virtual networks, and segmentation of the virtual network into subnets.

The following illustration depicts a virtual network that has a subnet containing two virtual machines. The virtual network has connections to an on-premises infrastructure and a separate virtual network.

Things to know about subnets

There are certain conditions for the IP addresses in a virtual network when you apply segmentation with subnets.

- Each subnet contains a range of IP addresses that fall within the virtual network address space.
- The address range for a subnet must be unique within the address space for the virtual network.
- The range for one subnet can't overlap with other subnet IP address ranges in the same virtual network.
- The IP address space for a subnet must be specified by using CIDR notation.
- You can segment a virtual network into one or more subnets in the Azure portal. Characteristics about the IP addresses for the subnets are listed.

+ Subnet | + Gateway subnet | Refresh | Manage users | Delete

Name	IPv4	IPv6	Available IPs	Delegated
subnet0	10.0.0.0/24	-	250	-
subnet1	10.0.1.0/24	-	251	-
subnet2	10.0.2.0/24	-	251	-
AzureBastionSubnet	10.0.30.0/26	-	26	-
GatewaySubnet	10.0.3.0/27	-	availability dependent on dynamic use	-

Reserved addresses

For each subnet, Azure reserves five IP addresses. The first four addresses and the last address are reserved.

Let's examine the reserved addresses in an IP address range of 192.168.1.0/24.

Reserved address	Reason
`192.168.1.0`	This value identifies the virtual network address.
`192.168.1.1`	Azure configures this address as the default gateway.
`192.168.1.2` *and* `192.168.1.3`	Azure maps these Azure DNS IP addresses to the virtual network space.
`192.168.1.255`	This value supplies the virtual network broadcast address.

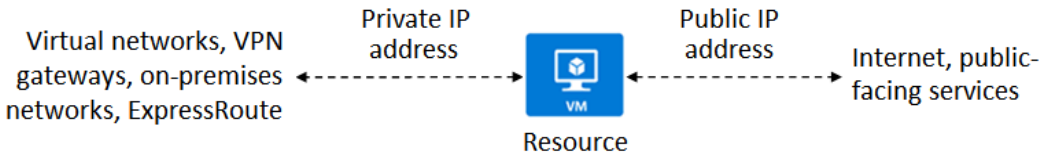
Plan IP addressing

You can assign IP addresses to Azure resources to communicate with other Azure resources, your on-premises network, and the internet. There are two types of Azure IP addresses: *private* and *public*.

Private IP addresses enable communication within an Azure virtual network and your on-premises network. You create a private IP address for your resource when you use a VPN gateway or Azure ExpressRoute circuit to extend your network to Azure.

Public IP addresses allow your resource to communicate with the internet. You can create a public IP address to connect with Azure public-facing services.

The following illustration shows a virtual machine resource that has a private IP address and a public IP address.



Private IP address assignment

A private IP address is allocated from the address range of the virtual network subnet that a resource is deployed in. There are two options: dynamic and static.

- **Dynamic**: Azure assigns the next available unassigned or unreserved IP address in the subnet's address range. Dynamic assignment is the default allocation method.

Suppose addresses 10.0.0.4 through 10.0.0.9 are already assigned to other resources. In this case, Azure assigns the address 10.0.0.10 to a new resource.

- **Static**: You select and assign any unassigned or unreserved IP address in the subnet's address range.

Suppose a subnet's address range is 10.0.0.0/16 and addresses 10.0.0.4 through 10.0.0.9 are already assigned to other resources. In this scenario, you can assign any address between 10.0.0.10 and 10.0.255.254.

Network Security Groups

Network security groups are a way to limit network traffic to resources in your virtual network. Network security groups contain a list of security rules that allow or deny inbound or outbound network traffic.

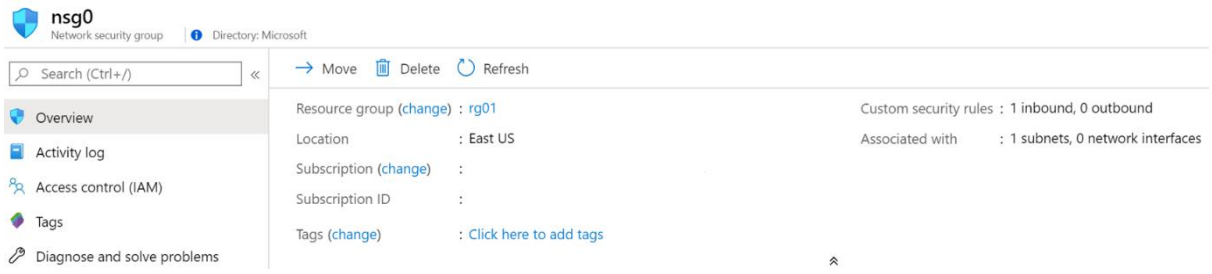
You can limit network traffic to resources in your virtual network by using a network security group. You can assign a network security group to a subnet or a network interface and define security rules in the group to control network traffic.

Things to know about network security groups

Let's look at the characteristics of network security groups.

- A network security group contains a list of security rules that allow or deny inbound or outbound network traffic.
- A network security group can be associated to a subnet or a network interface.
- A network security group can be associated multiple times.
- You create a network security group and define security rules in the Azure portal.

Network security groups are defined for your virtual machines in the Azure portal. The **Overview** page for a virtual machine provides information about the associated network security groups. You can see details such as the assigned subnets, assigned network interfaces, and the defined security rules.



Network security groups and subnets

You can assign network security groups to a subnet and create a protected screened subnet (also referred to as a demilitarized zone or *DMZ*). A DMZ acts as a buffer between resources within your virtual network and the internet.

- Use the network security group to restrict traffic flow to all machines that reside within the subnet.
- Each subnet can have a maximum of one associated network security group.

Network security groups and network interfaces

You can assign network security groups to a network interface card (NIC).

- Define network security group rules to control all traffic that flows through a network interface.
- Each network interface that exists in a subnet can have zero, or one, associated network security groups.

Things to know about security rules

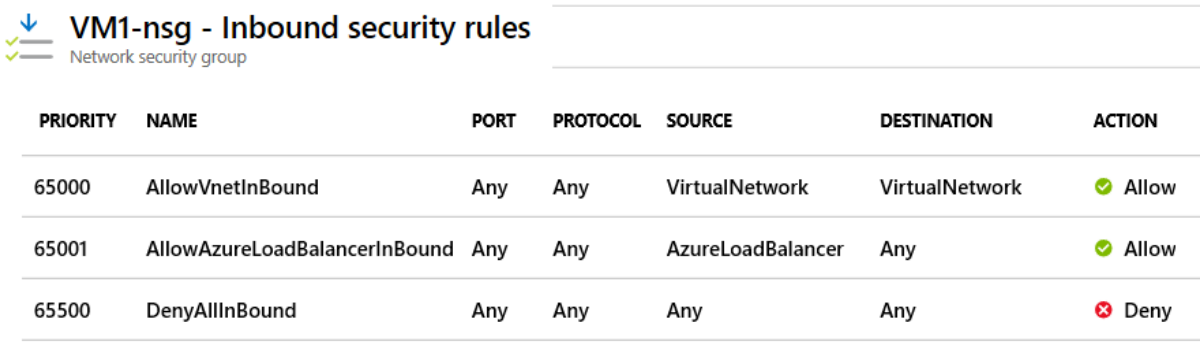
Let's review the characteristics of security rules in network security groups.

- Azure creates several default security rules within each network security group, including inbound traffic and outbound traffic. Examples of default rules include DenyAllInbound traffic and AllowInternetOutbound traffic.

- Azure creates the default security rules in each network security group that you create.
- You can add more security rules to a network security group by specifying conditions for any of the following settings:
    - **Name**
    - **Priority**
    - **Port**
    - **Protocol** (Any, TCP, UDP)
    - **Source** (Any, IP addresses, Service tag)
    - **Destination** (Any, IP addresses, Virtual network)
    - **Action** (Allow or Deny)
- Each security rule is assigned a Priority value. All security rules for a network security group are processed in priority order. When a rule has a low Priority value, the rule has a higher priority or precedence in terms of order processing.
- You can't remove the default security rules.
- You can override a default security rule by creating another security rule that has a higher Priority setting for your network security group.

Inbound traffic rules

Azure defines three default inbound security rules for your network security group. These rules **deny all inbound traffic** except traffic from your virtual network and Azure load balancers. The next image shows the default inbound security rules for a network security group in the Azure portal.



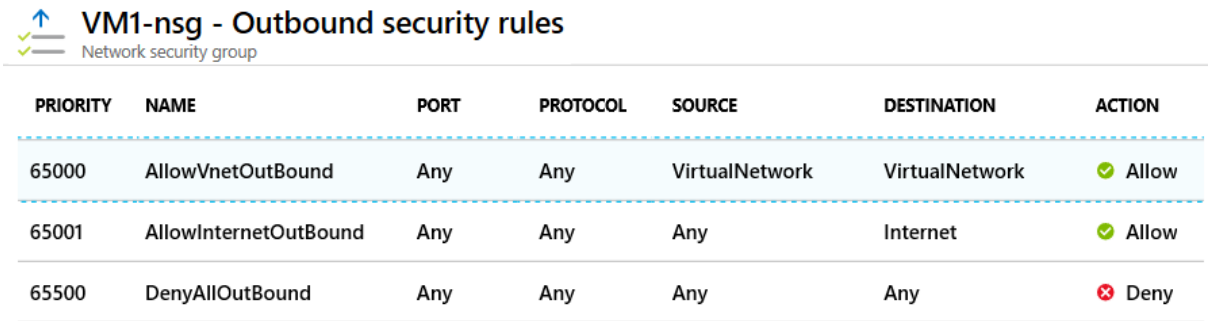
VM1-nsg - Inbound security rules

Network security group

PRIORITY	NAME	PORT	PROTOCOL	SOURCE	DESTINATION	ACTION
65000	AllowVnetInBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerInBound	Any	Any	AzureLoadBalancer	Any	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Outbound traffic rules

Azure defines three default outbound security rules for your network security group. These rules **only allow outbound traffic** to the internet and your virtual network. The next image shows the default outbound security rules for a network security group in the Azure portal.



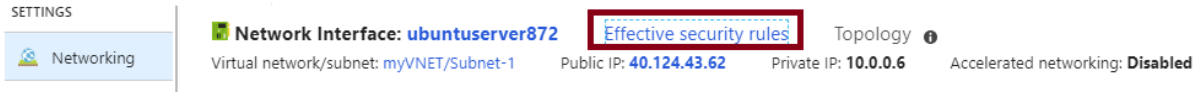
VM1-nsg - Outbound security rules

Network security group

PRIORITY	NAME	PORT	PROTOCOL	SOURCE	DESTINATION	ACTION
65000	AllowVnetOutBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowInternetOutBound	Any	Any	Any	Internet	Allow
65500	DenyAllOutBound	Any	Any	Any	Any	Deny

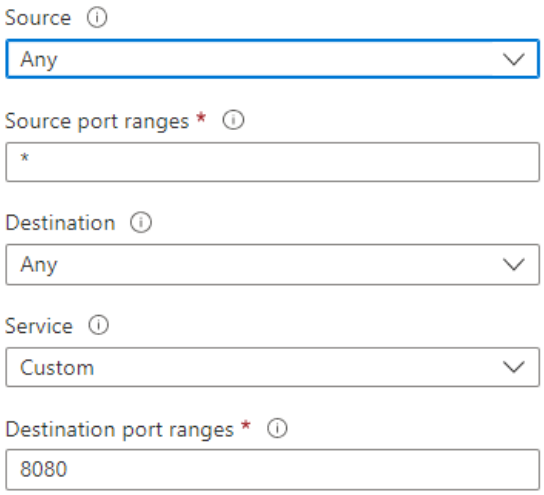
View effective security rules

If you have several network security groups and aren't sure which security rules are being applied, you can use the **Effective security rules** link in the Azure portal. You can use the link to verify which security rules are applied to your machines, subnets, and network interfaces.

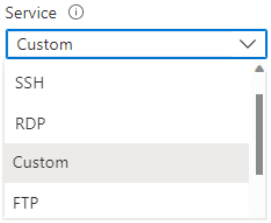


Things to know about configuring security rules

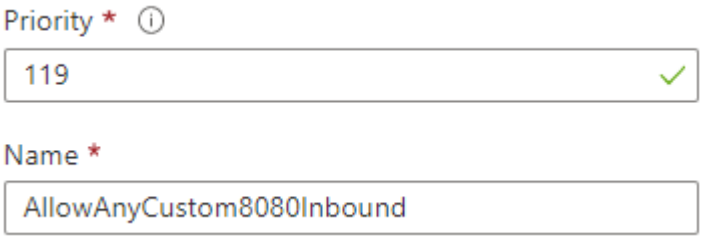
Let's look at some of the properties you need to specify to create your security rules. As you review these settings, think about the traffic rules you need to create and what services can fulfill your network requirements.



- **Source**: Identifies how the security rule controls **inbound** traffic. The value specifies a specific source IP address range to allow or deny. The source filter can be any resource, an IP address range, an application security group, or a default tag.
- **Destination**: Identifies how the security rule controls **outbound** traffic. The value specifies a specific destination IP address range to allow or deny. The destination filter value is similar to the source filter. The value can be any resource, an IP address range, an application security group, or a default tag.



- **Service**: Specifies the destination protocol and port range for the security rule. You can choose a predefined service like RDP or SSH or provide a custom port range. There are a large number of services to select from.
- **Priority**: Assigns the priority order value for the security rule. Rules are processed according to the priority order of all rules for a network security group, including a subnet and network interface. The lower the priority value, the higher priority for the rule.



Implement application security groups

You can implement application security groups in your Azure virtual network to logically group your virtual machines by workload. You can then define your network security group rules based on your application security groups.

Things to know about using application security groups

Application security groups work in the same way as network security groups, but they provide an application-centric way of looking at your infrastructure. You join your virtual machines to an application security group. Then you use the application security group as a source or destination in the network security group rules.

Let's examine how to implement application security groups by creating a configuration for an online retailer. In our example scenario, we need to control network traffic to virtual machines in application security groups.

Host your domain on Azure DNS

Azure DNS lets you host your Domain Name System (DNS) records for your domains on Azure infrastructure. With Azure DNS, you can use the same credentials, APIs, tools, and billing as your other Azure services.

What is Azure DNS?

Azure DNS is a hosting service for Domain Name System (DNS) domains that provides name resolution by using Microsoft Azure infrastructure.

What is DNS?

DNS, or the Domain Name System, is a protocol within the TCP/IP standard. DNS serves an essential role of translating the human-readable domain names—for example: www.wideworldimports.com—into a known IP address. IP addresses enable computers and network devices to identify and route requests among themselves.

DNS uses a global directory hosted on servers around the world. Microsoft is part of the network that provides a DNS service through Azure DNS.

A DNS server is also known as a DNS name server, or just a name server.

How does DNS work?

A DNS server carries out one of two primary functions:

- Maintains a local cache of recently accessed or used domain names and their IP addresses. This cache provides a faster response to a local domain lookup request. If the DNS server can't find the requested domain, it passes the request to another DNS server. This process repeats at each DNS server until either a match is made or the search times out.
- Maintains the key-value pair database of IP addresses and any host or subdomain over which the DNS server has authority. This function is often associated with mail, web, and other internet domain services.

DNS server assignment

In order for a computer, server, or other network-enabled device to access web-based resources, it must reference a DNS server.

When you connect by using your on-premises network, the DNS settings come from your server. When you connect by using an external location like a hotel, the DNS settings come from the internet service provider (ISP).

Domain lookup requests

Here's a simplified overview of the process a DNS server uses when it resolves a domain-name lookup request:

- If the domain name is stored in the short-term cache, the DNS server resolves the domain request.
- If the domain isn't in the cache, it contacts one or more DNS servers on the web to see if they have a match. When a match is found, the DNS server updates the local cache and resolves the request.
- If the domain isn't found after a reasonable number of DNS checks, the DNS server responds with a *domain cannot be found* error.

IPv4 and IPv6

Every computer, server, or network-enabled device on your network has an IP address. An IP address is unique within your domain. There are two standards of IP address: IPv4 and IPv6.

- **IPv4** is composed of four sets of numbers, in the range 0 to 255, each separated by a dot; for example: 127.0.0.1. Today, IPv4 is the most commonly used standard. Yet, with the increase in IoT devices, the IPv4 standard will eventually be unable to keep up.

- **IPv6** is a relatively new standard and is intended to eventually replace IPv4. It consists of eight groups of hexadecimal numbers, each separated by a colon; for example: fe80:11a1:ac15:e9gf:e884:edb0:ddee:fea3.

Many network devices are now provisioned with both an IPv4 and an IPv6 address. The DNS name server can resolve domain names to both IPv4 and IPv6 addresses.

DNS settings for your domain

Whether a third-party host your DNS server or you manage it in-house, you need to configure it for each host type you're using. Host types include web, email, or other services you're using.

As the administrator for your company, you want to set up a DNS server by using Azure DNS. In this instance, the DNS server acts as a start of authority (SOA) for your domain.

DNS record types

Configuration information for your DNS server is stored as a file within a zone on your DNS server. Each file is called a record. The following record types are the most commonly created and used:

- **A** is the host record, and is the most common type of DNS record. It maps the domain or host name to the IP address.
- **CNAME** is a Canonical Name record that's used to create an alias from one domain name to another domain name. If you had different domain names that all accessed the same website, you'd use CNAME.
- **MX** is the mail exchange record. It maps mail requests to your mail server, whether hosted on-premises or in the cloud.
- **TXT** is the text record. It's used to associate text strings with a domain name. Azure and Microsoft 365 use TXT records to verify domain ownership.

Additionally, there are the following record types:

- Wildcards
- CAA (certificate authority)
- NS (name server)
- SOA (start of authority)
- SPF (sender policy framework)
- SRV (server locations)

The SOA and NS records are created automatically when you create a DNS zone by using Azure DNS.

What is Azure DNS?

Azure DNS allows you to host and manage your domains by using a globally distributed name-server infrastructure. It allows you to manage all of your domains by using your existing Azure credentials.

Azure DNS acts as the SOA for the domain.

You can't use Azure DNS to register a domain name; you need to register it by using a third-party domain registrar.

Why use Azure DNS to host your domain?

Azure DNS is built on the Azure Resource Manager service, which offers the following benefits:

- Improved security
- Ease of use

- Private DNS domains
- Alias record sets

At this time, Azure DNS doesn't support Domain Name System Security Extensions. If you require this security extension, you should host those portions of your domain with a third-party provider.

Security features

Azure DNS provides the following security features:

- Role-based access control, which gives you fine-grained control over users' access to Azure resources. You can monitor their usage and control the resources and services to which they have access.
- Activity logs, which let you track changes to a resource and pinpoint where faults occurred.
- Resource locking, which gives you a greater level of control to restrict or remove access to resource groups, subscriptions, or any Azure resources.

Ease of use

Azure DNS can manage DNS records for your Azure services and provide DNS for your external resources. Azure DNS uses the same Azure credentials, support contract, and billing as your other Azure services.

You can manage your domains and records by using the Azure portal, Azure PowerShell cmdlets, or the Azure CLI. Applications that require automated DNS management can integrate with the service by using the REST API and software development kit (SDKs).

Private domains

Azure DNS handles translating external domain names to IP addresses. Azure DNS lets you create private zones. These zones provide name resolution for virtual machines (VMs) within a virtual network and between virtual networks without having to create a custom DNS solution. Private zones allow you to use your own custom domain names rather than the Azure-provided names.

To publish a private DNS zone to your virtual network, you specify the list of virtual networks that are allowed to resolve records within the zone.

Private DNS zones have the following benefits:

- DNS zones are supported as part of the Azure infrastructure, so there's no need to invest in a DNS solution.
- All DNS record types are supported: A, CNAME, TXT, MX, SOA, AAAA, PTR, and SRV.
- Host names for VMs in your virtual network are automatically maintained.
- Split-horizon DNS support allows the same domain name to exist in both private and public zones. It resolves to the correct one based on the originating request location.

Alias record sets

Alias records sets can point to an Azure resource. For example, you can set up an alias record to direct traffic to an Azure public IP address, an Azure Traffic Manager profile, or an Azure Content Delivery Network endpoint.

The alias record set is supported in the following DNS record types:

- A
- AAAA
- CNAME

Configure Azure Virtual Network peering

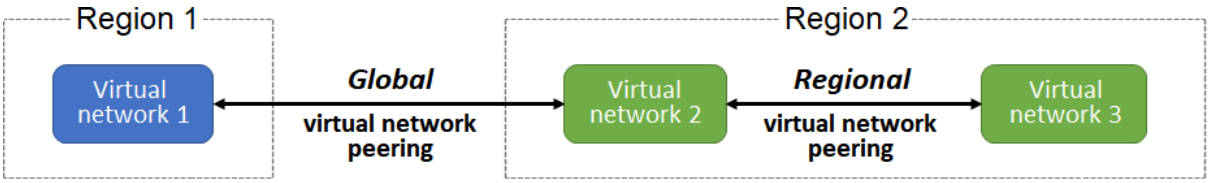
Azure Virtual Network peering lets you connect virtual networks in the same or different regions. Azure Virtual Network peering provides secure communication between resources in the peered networks.

Perhaps the simplest and quickest way to connect your virtual networks is to use Azure Virtual Network peering. Virtual Network peering enables you to seamlessly connect two Azure virtual networks. After the networks are peered, the two virtual networks operate as a single network, for connectivity purposes.

Things to know about Azure Virtual Network peering

Let's examine some prominent characteristics of Azure Virtual Network peering.

- There are two types of Azure Virtual Network peering: *regional* and *global*.



- **Regional virtual network peering** connects Azure virtual networks that exist in the same region.
- **Global virtual network peering** connects Azure virtual networks that exist in different regions.
- You can create a regional peering of virtual networks in the same Azure public cloud region, or in the same China cloud region, or in the same Microsoft Azure Government cloud region.
- You can create a global peering of virtual networks in any Azure public cloud region, or in any China cloud region.
- Global peering of virtual networks in different Azure Government cloud regions isn't permitted.
- After you create a peering between virtual networks, the individual virtual networks are still managed as separate resources.

Manage and control traffic flow in your Azure deployment with routes

To control traffic flow within your virtual network, you must learn the purpose and benefits of custom routes. You must also learn how to configure the routes to direct traffic flow through a network virtual appliance (NVA).

Azure routing

Network traffic in Azure is automatically routed across Azure subnets, virtual networks, and on-premises networks. System routes control this routing. They're assigned by default to each subnet in a virtual network. With these system routes, any Azure virtual machine that is deployed into a virtual network can communicate with any other in the network. These virtual machines are also potentially accessible from on-premises through a hybrid network or the internet.

You can't create or delete system routes, but you can override the system routes by adding custom routes to control traffic flow to the next hop.

Every subnet has the following default system routes:

Address prefix	Next hop type
Unique to the virtual network	Virtual network
0.0.0.0/0	Internet
10.0.0.0/8	None
172.16.0.0/12	None
192.168.0.0/16	None
100.64.0.0/10	None

The **Next hop type** column shows the network path taken by traffic sent to each address prefix. The path can be one of the following hop types:

- **Virtual network**: A route is created in the address prefix. The prefix represents each address range created at the virtual-network level. If multiple address ranges are specified, multiple routes are created for each address range.
- **Internet**: The default system route 0.0.0.0/0 routes any address range to the internet, unless you override Azure's default route with a custom route.
- **None**: Any traffic routed to this hop type is dropped and doesn't get routed outside the subnet. By default, the following IPv4 private-address prefixes are created: 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16. The prefix 100.64.0.0/10 for a shared address space is also added. None of these address ranges are globally routable.

The following diagram shows an overview of system routes and shows how traffic flows among subnets and the internet by default. You can see from the diagram that traffic flows freely among the two subnets and the internet.

Within Azure, there are other system routes. Azure creates these routes if the following capabilities are enabled:

- Virtual network peering
- Service chaining
- Virtual network gateway
- Virtual network service endpoint

Virtual network peering and service chaining

Virtual network peering and service chaining let virtual networks within Azure connect to one another. With this connection, virtual machines can communicate with each other within the same region or across regions. This communication in turn creates more routes within the default route table. Service chaining lets you override these routes by creating user-defined routes between peered networks.

The following diagram shows two virtual networks with peering configured. The user-defined routes are configured to route traffic through an NVA or an Azure VPN gateway.

Virtual network gateway

Use a virtual network gateway to send encrypted traffic between Azure and on-premises over the internet and to send encrypted traffic between Azure networks. A virtual network gateway contains routing tables and gateway services.

Virtual network service endpoint

Virtual network endpoints extend your private address space in Azure by providing a direct connection to your Azure resources. This connection restricts the flow of traffic: your Azure virtual machines can access your storage account

directly from the private address space and deny access from a public virtual machine. As you enable service endpoints, Azure creates routes in the route table to direct this traffic.

Custom routes

System routes might make it easy for you to quickly get your environment up and running. However, there are many scenarios in which you want to more closely control the traffic flow within your network. For example, you might want to route traffic through an NVA or through a firewall. This control is possible with custom routes.

You have two options for implementing custom routes: create a user-defined route, or use Border Gateway Protocol (BGP) to exchange routes between Azure and on-premises networks.

User-defined routes

You can use a user-defined route to override the default system routes so traffic can be routed through firewalls or NVAs.

For example, you might have a network with two subnets and want to add a virtual machine in the perimeter network to be used as a firewall. You can create a user-defined route so that traffic passes through the firewall and doesn't go directly between the subnets.

When creating user-defined routes, you can specify these next hop types:

- **Virtual appliance**: A virtual appliance is typically a firewall device used to analyze or filter traffic that is entering or leaving your network. You can specify the private IP address of a Network Interface Card (NIC) attached to a virtual machine so that IP forwarding can be enabled. Or you can provide the private IP address of an internal load balancer.
- **Virtual network gateway**: Use to indicate when you want routes for a specific address to be routed to a virtual network gateway. The virtual network gateway is specified as a VPN for the next hop type.
- **Virtual network**: Use to override the default system route within a virtual network.
- **Internet**: Use to route traffic to a specified address prefix that is routed to the internet.
- **None**: Use to drop traffic sent to a specified address prefix.

With user-defined routes, you can't specify the next hop type **VirtualNetworkServiceEndpoint**, which indicates virtual network peering.

Service tags for user-defined routes

You can specify a service tag as the address prefix for a user-defined route instead of an explicit IP range. A service tag represents a group of IP address prefixes from a given Azure service. Microsoft manages the address prefixes encompassed by the service tag and automatically updates the service tag as addresses change, thus minimizing the complexity of frequent updates to user-defined routes and reducing the number of routes you need to create.

Border gateway protocol

A network gateway in your on-premises network can exchange routes with a virtual network gateway in Azure by using BGP. BGP is the standard routing protocol that's normally used to exchange routing information among two or more networks. BGP is used to transfer data and information between autonomous systems on the internet, such as different host gateways.

Typically, you use BGP to advertise on-premises routes to Azure when you're connected to an Azure datacenter through Azure ExpressRoute. You can also configure BGP if you connect to an Azure virtual network by using a VPN site-to-site connection.

Route selection and priority

If multiple routes are available in a route table, Azure uses the route with the longest prefix match. For example, a message is sent to the IP address 10.0.0.2, but two routes are available with the 10.0.0.0/16 and 10.0.0.0/24 prefixes. Azure selects the route with the 10.0.0.0/24 prefix because it's more specific.

The longer the route prefix, the shorter the list of IP addresses available through that prefix. When you use longer prefixes, the routing algorithm can select the intended address more quickly.

You can't configure multiple user-defined routes with the same address prefix.

If there are multiple routes with the same address prefix, Azure selects the route based on the type in the following order of priority:

1. User-defined routes
2. BGP routes
3. System routes

What is an NVA?

A network virtual appliance (NVA) is a virtual appliance that consists of various layers like:

- A firewall
- A WAN optimizer
- Application-delivery controllers
- Routers
- Load balancers
- IDS/IPS
- Proxies

You can deploy NVAs that you choose from providers in Azure Marketplace. Such providers include Cisco, Check Point, Barracuda, Sophos, WatchGuard, and SonicWall. You can use an NVA to filter traffic inbound to a virtual network, to block malicious requests, and to block requests made from unexpected resources.

In the retail-organization example scenario, you must work with the security and network teams. You want to implement a secure environment that scrutinizes all incoming traffic and blocks unauthorized traffic from passing on to the internal network. You also want to secure both virtual-machine networking and Azure-services networking as part of your company's network-security strategy.

Your goal is to prevent unwanted or unsecured network traffic from reaching key systems.

As part of the network-security strategy, you must control the flow of traffic within your virtual network. You also must learn the role of an NVA and the benefit of using an NVA to control traffic flow through an Azure network.

Network virtual appliance

Network virtual appliances (NVAs) are virtual machines that control the flow of network traffic by controlling routing. You'll typically use them to manage traffic flowing from a perimeter-network environment to other networks or subnets.

You can deploy firewall appliances into a virtual network in different configurations. You can put a firewall appliance in a perimeter-network subnet in the virtual network, or if you want more control of security, implement a microsegmentation approach.

With the microsegmentation approach, you can create dedicated subnets for the firewall and then deploy web applications and other services in other subnets. All traffic is routed through the firewall and inspected by the NVAs. You'll enable forwarding on the virtual-appliance network interfaces to pass traffic that is accepted by the appropriate subnet.

Microsegmentation lets the firewall inspect all packets at OSI Layer 4 and, for application-aware appliances, Layer 7. When you deploy an NVA to Azure, it acts as a router that forwards requests between subnets on the virtual network.

Some NVAs require multiple network interfaces. One network interface is dedicated to the management network for the appliance. Additional network interfaces manage and control the traffic processing. After you've deployed the NVA, you can then configure the appliance to route the traffic through the proper interface.

User-defined routes

For most environments, the default system routes already defined by Azure are enough to get the environments up and running. In certain cases, you should create a routing table and add custom routes. Examples include:

- Access to the internet via on-premises network using forced tunneling

- Using virtual appliances to control traffic flow

You can create multiple route tables in Azure. Each route table can be associated with one or more subnets. A subnet can only be associated with one route table.

Network virtual appliances in a highly available architecture

If traffic is routed through an NVA, the NVA becomes a critical piece of your infrastructure. Any NVA failures directly affect the ability of your services to communicate. It's important to include a highly available architecture in your NVA deployment.

Introduction to Azure Load Balancer

Azure Load Balancer is an Azure service that allows you to evenly distribute incoming network traffic across a group of Azure VMs, or across instances in a Virtual Machine Scale Set. Load Balancer delivers high availability and network performance in the following ways:

- Load-balancing rules determine how traffic is distributed to instances that comprise the back end.
- Health probes ensure the resources in the back end are healthy and that traffic isn't directed to unhealthy back-end instances.

You can deploy **public** load balancers and **internal** (or *private*) load balancers in Azure:

- *Public load balancers* are used to load balance internet traffic to your VMs. A public load balancer maps the public IP address and port number of incoming traffic to the private IP address and port number of the back-end pool VMs. For example, you can spread the load of incoming web-request traffic from the internet across multiple web servers. A public load balancer can also provide outbound connections for VMs inside your virtual network.
- An *internal load balancer* directs traffic to resources that are inside a virtual network or that use a VPN to access Azure infrastructure. Internal load balancer front-end IP addresses and virtual networks are never directly exposed to an internet endpoint. Internal line-of-business (LOB) applications run in Azure and are accessed from within Azure or from on-premises resources. An internal load balancer is used where private IPs are needed at the front end only. Internal load balancers are often used to balance traffic from the front-end web tier infrastructure as a service (IaaS) VMs across a set of secondary VMs that perform tasks such as performing calculations or data processing.

An internal load balancer enables the following types of load balancing:

- **Within a virtual network**: Load balancing from VMs in the virtual network to a set of VMs that reside within the same virtual network.
- **For a cross-premises virtual network**: Load balancing from on-premises computers to a set of VMs that reside within the same virtual network.
- **For multi-tier applications**: Load balancing for internet-facing multi-tier applications where the back-end tiers aren't internet-facing. The back-end tiers require traffic load balancing from the internet-facing tier.
- **For LOB applications**: Load balancing for LOB applications that are hosted in Azure without added load balancer hardware or software. This scenario includes on-premises servers that are in the set of computers whose traffic is load balanced.

Each Load Balancer type can be used for inbound and outbound scenarios and scale up to millions of TCP and UDP application flows.

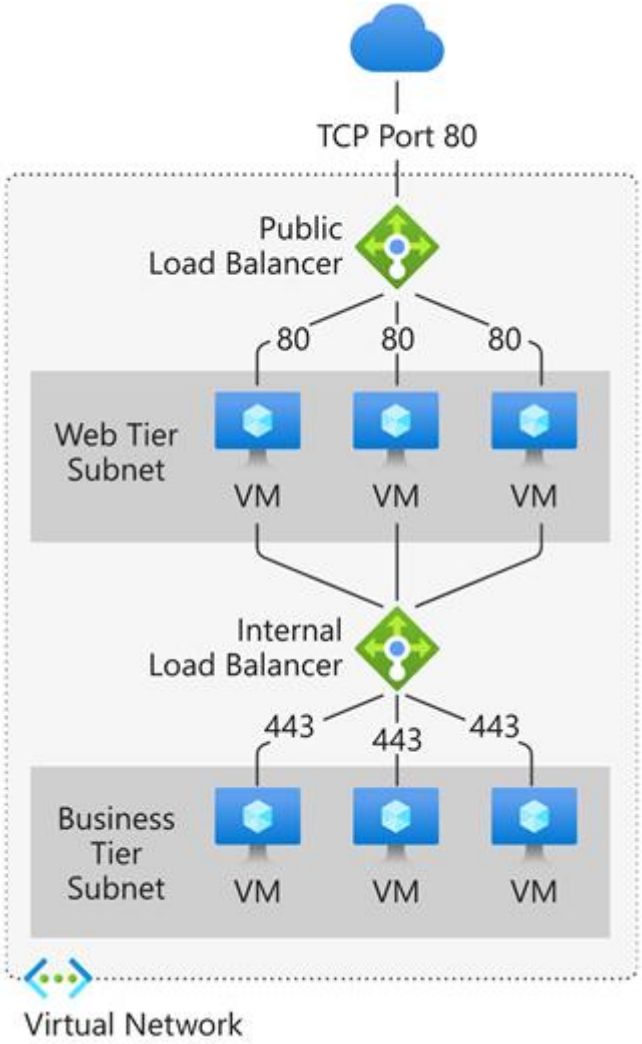
How Azure Load Balancer works

Azure Load Balancer operates at the transport layer of the OSI model. This Layer 4 functionality allows traffic management based on specific properties of the traffic. Properties including, source and destination address, TCP or UDP protocol type, and port number.

Load Balancer has several elements that work together to ensure an application's high availability and performance:

- Front-end IP
- Load balancer rules
- Back-end pool

- Health probes
- Inbound NAT rules
- High availability ports
- Outbound rules



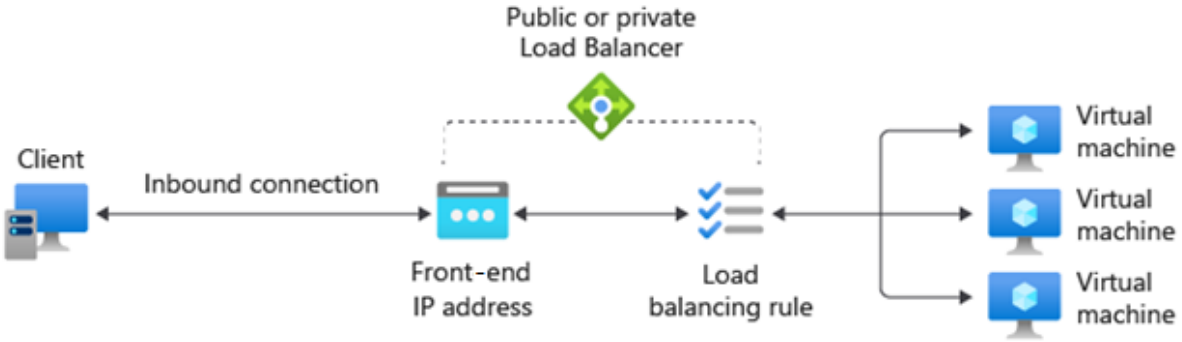
Front-end IP

The front-end IP address is the address clients use to connect to your web application. A front-end IP address can be either a public or a private IP address. Azure load balancers can have multiple front-end IPs. The selection of a public or a private IP address determines which type of load balancer to create:

- **Public IP address: A public load balancer**: A public load balancer maps the public IP and port of incoming traffic to the private IP and port of the VM. You can distribute specific types of traffic across multiple VMs or services by applying load-balancing rules. For example, you can spread the load of web request traffic across multiple web servers. The load balancer maps the response traffic from the private IP and port of the VM to the public IP and port of the load balancer. Then, it transmits the response back to the requesting client.
- **Private IP address: An internal load balancer**: An internal load balancer distributes traffic to resources that are inside a virtual network. Azure restricts access to the front-end IP addresses of a virtual network that are load balanced. Front-end IP addresses and virtual networks are never directly exposed to an internet endpoint. Internal line-of-business applications run in Azure and are accessed from within Azure or from on-premises resources through a VPN or ExpressRoute connection.

Load Balancer rules

A load balancer rule defines how traffic is distributed to the back-end pool. The rule maps a given front-end IP and port combination to a set of back-end IP addresses and port combination.



Traffic is managed using a five-tuple hash made from the following elements:

- **Source IP**: The IP address of the requesting client.
- **Source port**: The port of the requesting client.
- **Destination IP**: The destination IP address of the request.

- **Destination port**: The destination port of the request.
- **Protocol type**: The specified protocol type, TCP or UDP.
- **Session affinity**: Ensures that the same pool node always handles traffic for a client.

Load Balancer allows you to load balance services on multiple ports, multiple IP addresses, or both. You can configure different load balancing rules for each front-end IP. Multiple front-end configurations are only supported with IaaS VMs.

Load Balancer can't apply different rules based on internal traffic content because it operates at Layer 4 (transport layer) of the OSI model. If you need to manage traffic based on its Layer 7 (application layer) properties, you need to deploy a solution like Azure Application Gateway.

Back-end pool

The back-end pool is a group of VMs or instances in a Virtual Machine Scale Set that responds to the incoming request. To scale cost-effectively to meet high volumes of incoming traffic, computing guidelines generally recommend adding more instances to the back-end pool.

Load Balancer implements automatic reconfiguration to redistribute load across the altered number of instances when you scale instances up or down. For example, if you added two more VMs instances to the back-end pool, Load Balancer would reconfigure itself to start balancing traffic to those instances based on the already configured load balancing rules.

Health probes

A health probe is used to determine the health status of the instances in the back-end pool. This health probe determines if an instance is healthy and can receive traffic. You can define the unhealthy threshold for your health probes. When a probe fails to respond, the load balancer stops sending new connections to the unhealthy instances. A probe failure doesn't affect existing connections. The connection continues until:

- The application ends the flow.
- Idle timeout occurs.
- The VM shuts down.

Load Balancer allows you to configure different health probe types for endpoints: TCP, HTTP, and HTTPS.

- **TCP custom probe**: This probe relies on establishing a successful TCP session to a defined probe port. If the specified listener on the VM exists, the probe succeeds. If the connection is refused, the probe fails. You can specify the Port, Interval, and Unhealthy threshold.
- **HTTP or HTTPS custom probe**: The load balancer regularly probes your endpoint (every 15 seconds, by default). The instance is healthy if it responds with an HTTP 200 within the timeout period (default of 31 seconds). Any status other than HTTP 200 causes the probe to fail. You can specify the port (Port), the URI for requesting the health status from the back end (URI), amount of time between probe attempts (Interval), and the number of failures that must occur for the instance to be considered unhealthy (Unhealthy threshold).

Session persistence

By default, Load Balancer distributes network traffic equally among multiple VM instances. It provides stickiness only within a transport session. Session persistence specifies how traffic from a client should be handled. The default behavior (None) is that any healthy VM can handle successive requests from a client.

Session persistence is also known as session affinity, source IP affinity, or client IP affinity. This distribution mode uses a two-tuple (source IP and destination IP) or three-tuple (source IP, destination IP, and protocol type) hash to route to

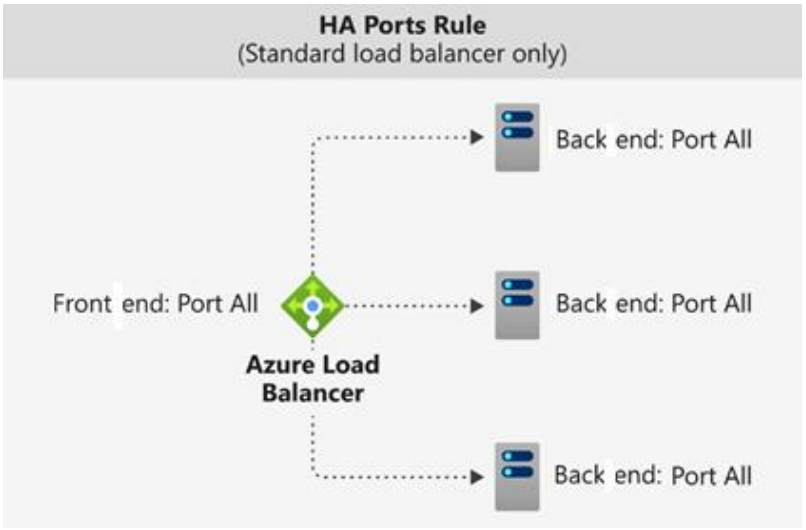
back-end instances. When you use session persistence, connections from the same client go to the same back-end instance within the back-end pool. You can configure one of the following session persistence options:

- **None (default)**: Specifies that any healthy VM can handle the request.
- **Client IP (2-tuple)**: Specifies that the same back-end instance can handle successive requests from the same client IP address.
- **Client IP and protocol (3-tuple)**: Specifies that the same back-end instance can handle successive requests from the same client IP address and protocol combination.

You can change this behavior by configuring one of the options that are described in the following sections.

High availability ports

A load balancer rule configured with protocol - all and port - 0 is called a *high availability (HA) port rule*. This rule enables a single rule to load balance all TCP and UDP flows that arrive on all ports of an internal standard load balancer.



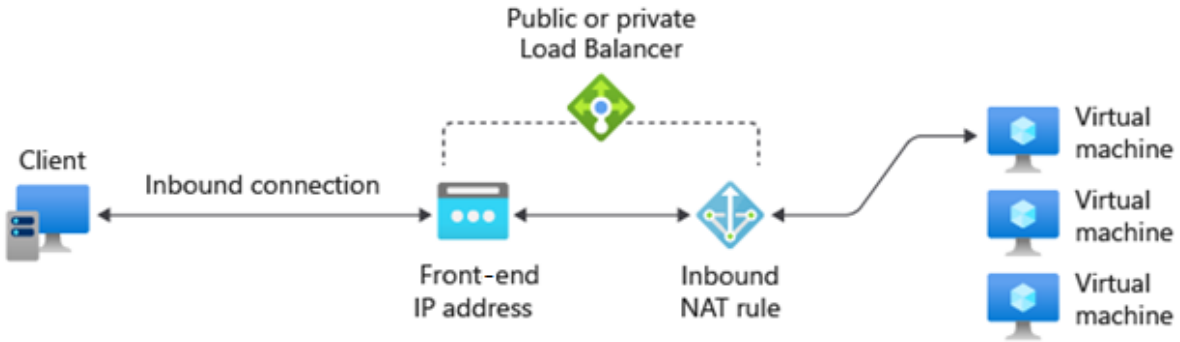
The load-balancing decision is made per flow. This action is based on the following five-tuple connection:

- Source IP address
- Source port
- Destination IP address
- Destination port
- Protocol

HA ports load-balancing rules help you with critical scenarios, such as high availability and scale for network virtual appliances (NVAs) inside virtual networks. The feature can help when a large number of ports must be load balanced.

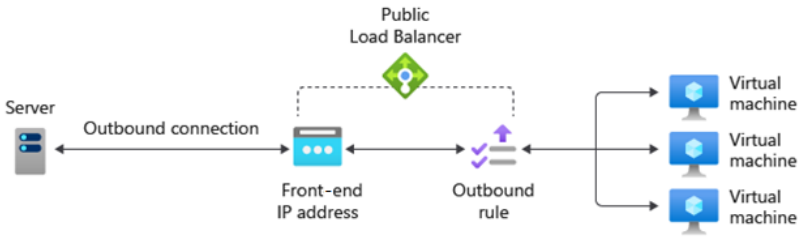
Inbound NAT rules

You can use load balancing rules in combination with Network Address Translation (NAT) rules. For example, you could use NAT from the load balancer's public address to TCP 3389 on a specific VM. This rule combination allows remote desktop access from outside of Azure.



Outbound rules

An outbound rule configures Source Network Address Translation (SNAT) for all VMs or instances identified by the back-end pool. This rule enables instances in the back end to communicate (outbound) to the internet or other public endpoints.



When to use Azure Load Balancer

Azure Load Balancer is best suited for applications that require ultra-low latency and high performance. Load Balancer is suitable for your organization's needs because you're replacing existing network hardware devices that load balance traffic across applications. The applications used multiple VM tiers when the applications were on-premises with an Azure service that has the same functionality.

Because Load Balancer operates at Layer 4 like hardware devices that were used on-premises before the organization migrated to Azure, you can use Load Balancer to replicate that hardware device functionality. This functionality includes using health probes to ensure that Load Balancer doesn't forward traffic to failed VM nodes. It also includes using session persistence to ensure that clients only communicate with a single VM during a session.

When not to use Azure Load Balancer

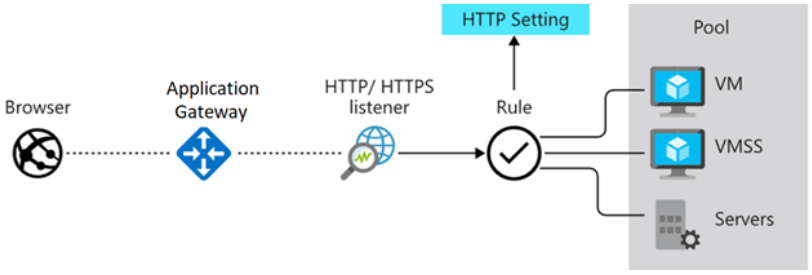
Azure Load Balancer isn't appropriate if you have a web application that doesn't require load balancing running on a single IaaS VM instance.

Azure provides other load-balancing solutions as alternatives to Azure Load Balancer, including Azure Front Door, Azure Traffic Manager, and Azure Application Gateway:

- **Azure Front Door** is an application-delivery network that provides a global load balancing and site acceleration service for web applications. It offers Layer 7 capabilities for your application like TLS/SSL offload, path-based routing, fast failover, a web application firewall, and caching to improve performance and high availability of your applications.
- **Azure Traffic Manager** is a DNS-based traffic load balancer that allows you to distribute traffic optimally to services across global Azure regions while providing high availability and responsiveness. Because Traffic Manager is a DNS-based load-balancing service, it load balances only at the domain level. For that reason, it can't fail over as quickly as Front Door, because of common challenges around DNS caching and systems not honoring DNS TTLs.
- **Azure Application Gateway** provides Application Delivery Controller (ADC) as a service, offering various Layer 7 load-balancing capabilities. Use it to optimize web farm productivity by offloading CPU-intensive TLS/SSL termination to the gateway. Application Gateway works within a region rather than globally.
- **Azure Load Balancer** is a high-performance, ultra-low-latency Layer 4 load-balancing service (inbound and outbound) for all UDP and TCP protocols. Its built to handle millions of requests per second while ensuring your solution is highly available. Azure Load Balancer is zone-redundant, ensuring high availability across availability zones. If Adatum had applications that required web application firewall functionality, Azure Load Balancer wouldn't be an appropriate solution for the company.

What is Azure Application Gateway?

Azure Application Gateway manages the requests that client applications send to web apps that are hosted on a pool of web servers. The pool of web servers can be Azure virtual machines, Azure Virtual Machine Scale Sets, Azure App Service, and even on-premises servers.



Application Gateway provides features such as load balancing HTTP traffic and web application firewall. It provides support for TLS/SSL encryption of traffic between users and an application gateway and between application servers and an application gateway.

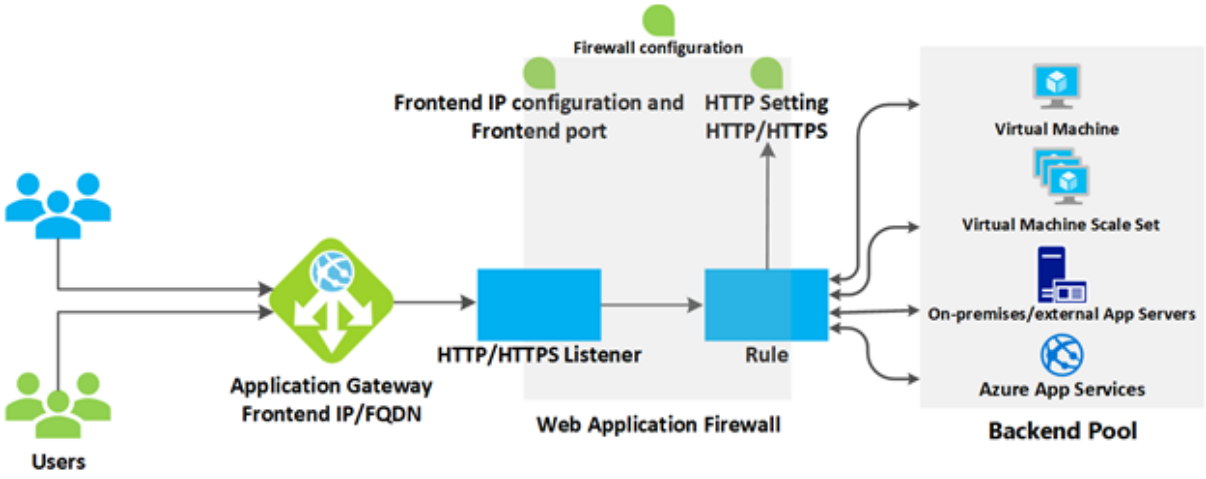
Application Gateway uses a round-robin process to load balance requests to the servers in each back-end pool. Session stickiness ensures client requests in the same session are routed to the same back-end server. Session stickiness is especially important with e-commerce applications where you don't want a transaction to be disrupted because the load balancer bounces it around between back-end servers.

Azure Application Gateway includes the following features:

- Support for the HTTP, HTTPS, HTTP/2, and WebSocket protocols
- A web application firewall to protect against web application vulnerabilities
- End-to-end request encryption
- Autoscaling to dynamically adjust capacity as your web traffic load change
- Connection draining allowing graceful removal of back-end pool members during planned service updates

How Azure Application Gateway works

Azure Application Gateway has a series of components that combine to securely route and load balance requests across a pool of web servers. Application Gateway includes the following components:



- **Front-end IP address**: Client requests are received through a front-end IP address. You can configure Application Gateway to have a public IP address, a private IP address, or both. Application Gateway can't have more than one public IP address and one private IP address.
- **Listeners**: Application Gateway uses one or more listeners to receive incoming requests. A listener accepts traffic arriving on a specified combination of protocol, port, host, and IP address. Each listener routes requests

to a back-end pool of servers following routing rules that you specify. A listener can be Basic or Multi-site. A *Basic* listener only routes a request based on the path in the URL. A *Multi-site* listener can also route requests using the hostname element of the URL. Listeners also handle TLS/SSL certificates for securing your application between the user and Application Gateway.

- **Routing rules**: A routing rule binds a listener to the back-end pools. A rule specifies how to interpret the hostname and path elements in the URL of a request and direct the request to the appropriate back-end pool. A routing rule also has an associated set of HTTP settings. These HTTP settings indicate whether (and how) traffic is encrypted between Application Gateway and the back-end servers. Other configuration information includes Protocol, Session stickiness, Connection draining, Request timeout period, and Health probes.

Load balancing in Application Gateway

Application Gateway automatically load balances the requests sent to the servers in each back-end pool using a round-robin mechanism. Load-balancing works with the OSI Layer 7 routing implemented by Application Gateway routing, which means that it load balances requests based on the routing parameters (host names and paths) used by the Application Gateway rules. In comparison, other load balancers, such as Azure Load Balancer, function at the OSI Layer 4 level and distribute traffic based on the IP address of the target of a request.

You can configure session stickiness if you need to ensure that all requests for a client in the same session are routed to the same server in a back-end pool.

Web application firewall

The web application firewall (WAF) is an optional component that handles incoming requests before they reach a listener. The web application firewall checks each request for many common threats based on the Open Web Application Security Project (OWASP). Common threats include: SQL-injection, Cross-site scripting, Command injection, HTTP request smuggling, HTTP response splitting, Remote file inclusion, Bots, crawlers, and scanners, and HTTP protocol violations and anomalies.

OWASP defines a set of generic rules for detecting attacks. These rules are referred to as the Core Rule Set (CRS). The rule sets are under continuous review as attacks evolve in sophistication. WAF supports four rule sets: CRS 3.2, 3.1, 3.0 and 2.2.9. CRS 3.1 is the default. If necessary, you can opt to select only specific rules in a rule set, targeting certain threats. Additionally, you can customize the firewall to specify which elements in a request to examine, and limit the size of messages to prevent massive uploads from overwhelming your servers.

Back-end pools

A back-end pool is a collection of web servers that can be made up of: a fixed set of virtual machines, a virtual machine scale-set, an app hosted by Azure App Services, or a collection of on-premises servers.

Each back-end pool has an associated load balancer that distributes work across the pool. When configuring the pool, you provide the IP address or name of each web server. All the servers in the back-end pool should be configured in the same way, including their security settings.

If you're using TLS/SSL, the back-end pool has an HTTP setting that references a certificate used to authenticate the back-end servers. The gateway re-encrypts the traffic by using this certificate before sending it to one of your servers in the back-end pool.

If you're using Azure App Service to host the back-end application, you don't need to install any certificates in Application Gateway to connect to the back-end pool. All communications are automatically encrypted. Application Gateway trusts the servers because Azure manages them.

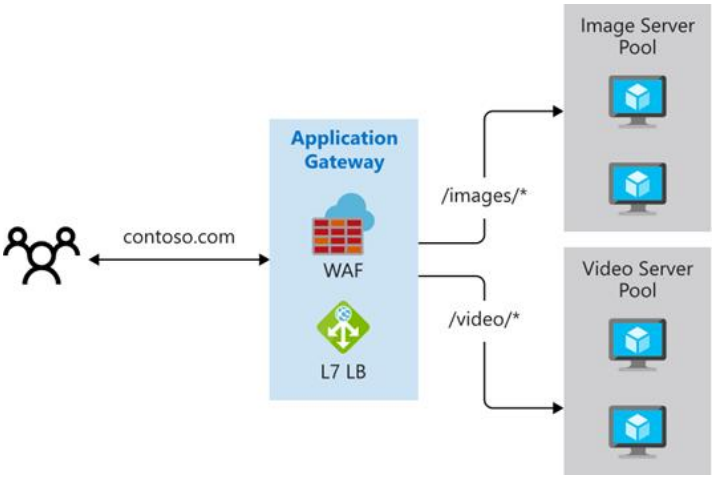
Application Gateway uses a rule to specify how to direct the messages that it receives on its incoming port to the servers in the back-end pool. If the servers are using TLS/SSL, you must configure the rule to indicate:

- That your servers expect traffic through the HTTPS protocol.

- Which certificate to use to encrypt traffic and authenticate the connection to a server.

Application Gateway routing

When the gateway routes a client request to a web server in the back-end pool, it uses a set of rules configured for the gateway to determine where the request should go. There are two primary methods of routing this client request traffic: path-based routing and multiple-site routing.

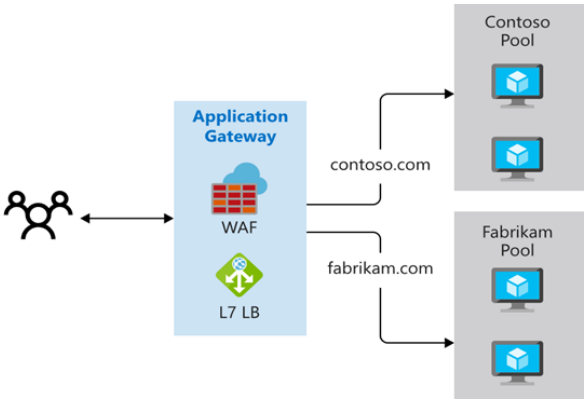


Path-based routing

Path-based routing sends requests with different URL paths to different pools of back-end servers. For example, you could direct requests with the path /video/* to a back-end pool containing servers that are optimized to handle video streaming, and direct /images/* requests to a pool of servers that handle image retrieval.

Multiple-site routing

Multiple-site routing configures more than one web application on the same Application Gateway instance. In a multi-site configuration, you register multiple DNS names (CNAMEs) for the IP address of the application gateway, specifying the name of each site. Application Gateway uses separate listeners to wait for requests for each site. Each listener passes the request to a different rule, which can route the requests to servers in a different back-end pool. For example, you could direct all requests for http://contoso.com to servers in one back-end pool, and requests for http://fabrikam.com to another back-end pool. The following diagram shows this configuration:



Multi-site configurations are useful for supporting multitenant applications, where each tenant has its own set of virtual machines or other resources hosting a web application.

Application Gateway routing also includes these features:

- **Redirection**. Redirection can be used to another site, or from HTTP to HTTPS.
- **Rewrite HTTP headers**. HTTP headers allow the client and server to pass parameter information with the request or the response.
- **Custom error pages**. Application Gateway allows you to create custom error pages instead of displaying default error pages. You can use your own branding and layout using a custom error page.

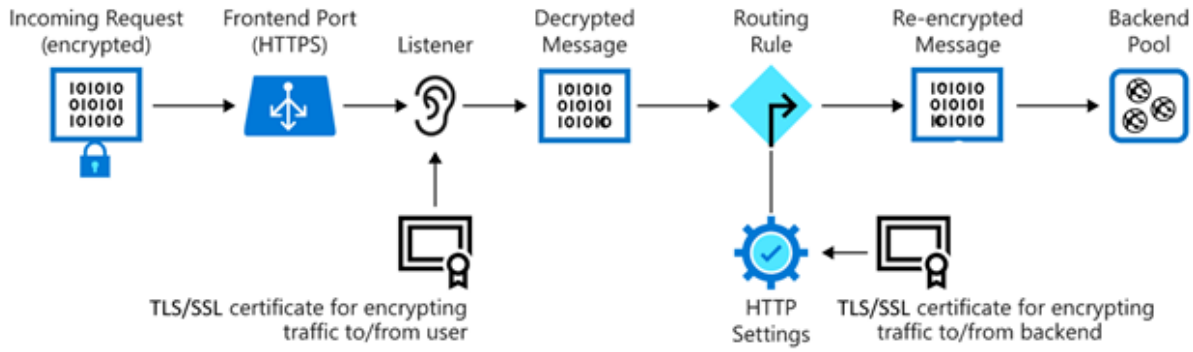
TLS/SSL termination

When you terminate the TLS/SSL connection at the application gateway, it offloads the CPU-intensive TLS/SSL termination workload from your servers. Also, you don't need to install certificates and configure TLS/SSL on your servers.

If you need end-to-end encryption, Application Gateway can decrypt the traffic on the gateway by using your private key, then re-encrypt again with the public key of the service running in the back-end pool.

Traffic enters the gateway through a front-end port. You can open many ports, and Application Gateway can receive messages on any of these ports. A listener is the first thing that your traffic meets when entering the gateway through a port. The listener is set up to listen for a specific host name, and a specific port on a specific IP address. The listener

can use an TLS/SSL certificate to decrypt the traffic that enters the gateway. The listener then uses a rule that you define to direct the incoming requests to a back-end pool.



Exposing your website or web application through the application gateway also means that you don't directly connect your servers to the web. You're exposing only port 80 or port 443 on the application gateway, which is then forwarded to the back-end pool server. In this configuration, your web servers aren't directly accessible from the internet, which reduces the attack surface of your infrastructure.

Health probes

Health probes determine which servers are available for load-balancing in a back-end pool. The Application Gateway uses a health probe to send a request to a server. When the server returns an HTTP response with a status code between 200 and 399, the server is considered healthy. If you don't configure a health probe, Application Gateway creates a default probe that waits for 30 seconds before deciding that a server is unavailable. Health probes ensure that traffic isn't directed to a nonresponsive or failed web endpoint in the back-end pool.

Autoscaling

Application Gateway supports autoscaling, and can scale up or down based on changing traffic load patterns. Autoscaling also removes the requirement to choose a deployment size or instance count during provisioning.

WebSocket and HTTP/2 traffic

Application Gateway provides native support for the WebSocket and HTTP/2 protocols. The WebSocket and HTTP/2 protocols enable full duplex communication between a server and a client over a long-running TCP connection. This type of communication is more interactive between the web server and the client, and can be bidirectional without the need for polling as required in HTTP-based implementations. These protocols have low overhead (unlike HTTP) and can reuse the same TCP connection for multiple request/responses resulting in a more efficient resource utilization. These protocols are designed to work over traditional HTTP ports of 80 and 443.

When to use Azure Application Gateway

Azure Application Gateway can meet your organization’s needs for the following reasons:

- Azure Application Gateway routing allows traffic to be directed from an endpoint in Azure to a back-end pool made up of servers running in Adatum’s on-premises datacenter. The health-probe functionality of Azure Application Gateway ensures that traffic isn't being directed to any server that becomes unavailable.
- Azure Application Gateway TLS termination functionality reduces the amount of CPU capacity that servers in the back-end pool allocate to encryption and decryption operations.
- Azure Application Gateway allows Adatum to use a web application firewall to block cross-site scripting and SQL injection traffic before it reaches servers in the back-end pool.

- Azure Application Gateway supports session affinity. This support is required because the several web applications deployed by Adatum use user session state information stored locally on individual servers in the back-end pool.

When not to use Azure Application Gateway

Azure Application Gateway isn't appropriate if you have a web application that doesn't require load balancing. For example, if you have a web application that only receives a small amount of traffic and the existing infrastructure already competently deals with the existing load, there's no need to deploy a back-end pool of web apps or virtual machines and no need for Application Gateway.

Azure provides other load balancing solutions, including Azure Front Door, Azure Traffic Manager, and Azure Load Balancer. The following list describes the differences between these services:

- **Front Door** is an application delivery network that provides global load balancing and site acceleration service for web applications. It offers Layer 7 capabilities for your application like TLS/SSL offload, path-based routing, fast failover, web application firewall, and caching to improve performance and high-availability of your applications. Choose this option in scenarios such as load balancing a web app deployed across multiple Azure regions.
- **Traffic Manager** is a DNS-based traffic load balancer that enables you to distribute traffic optimally to services across global Azure regions while providing high availability and responsiveness. Because Traffic Manager is a DNS-based load-balancing service, it load-balances only at the domain level. For that reason, it can't fail over as quickly as Front Door because of common challenges around DNS caching and systems not honoring DNS TTLs.
- **Azure Load Balancer** is a high-performance, ultra low-latency Layer 4 load-balancing service (inbound and outbound) for all UDP and TCP protocols. Azure Load Balancer is built to handle millions of requests per second while ensuring that your solution is highly available. Azure Load Balancer is zone-redundant, ensuring high availability across availability zones. Azure Load Balancer works within a region rather than globally.

Traffic Manager

Your customers require 24/7 availability of your company's streaming music application. Cloud services in one region might become unavailable because of technical issues such as planned maintenance or scheduled security updates. In these scenarios, your company wants to have a failover endpoint so your customers can continue to access its services. To manage routing traffic and to handle these situations, you've decided to implement Azure Traffic Manager.

How Traffic Manager works

When a client attempts to connect to a service, first it resolves the DNS name of the service as an IP address. The client then connects to that IP address to access the service.

Traffic Manager uses DNS to direct clients to a specific service endpoint IP address based on the rules of the traffic routing method that's used. Clients connect directly to the selected endpoint. Traffic Manager isn't a proxy or gateway. Traffic Manager doesn't see the traffic that passes between the clients and the service; it just gives clients the IP address of where they need to go.

Traffic Manager endpoints

An endpoint is the destination location that's returned to the client. You'll configure each application deployment as an 'endpoint' in Traffic Manager. When Traffic Manager receives a DNS request, it chooses an available endpoint to return in the DNS response. There are three types of endpoint Traffic Manager supports:

- **Azure endpoints** are used for services hosted in Azure. These can be services like Azure App Service, and public IP resources that are associated with load balancers or virtual machines.
- **External endpoints** are used for IPv4/IPv6 addresses, FQDNs, or for services hosted outside Azure either on-premises or with a different hosting provider.
- **Nested endpoints** are used to combine Traffic Manager profiles to create more flexible traffic-routing schemes to support the needs of larger, more complex deployments.

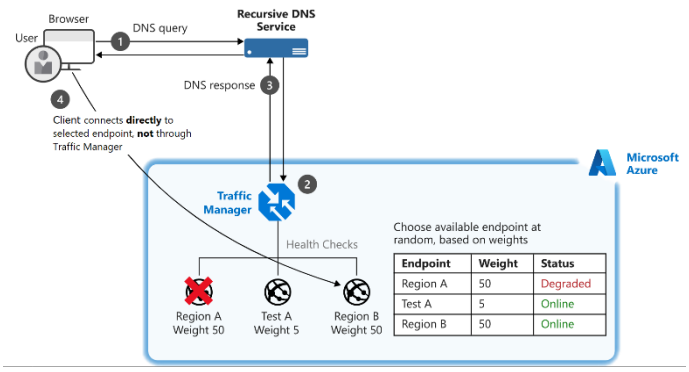
There's no restriction about how endpoints of different types are combined in a single Traffic Manager profile. Each profile can contain any mix of endpoint types.

Traffic Manager routing methods

Traffic Manager supports different methods for choosing how traffic is routed to multiple endpoints. Traffic Manager applies a traffic-routing method to each DNS query it receives and determines which endpoint is returned in the response. You can choose from six traffic routing methods.

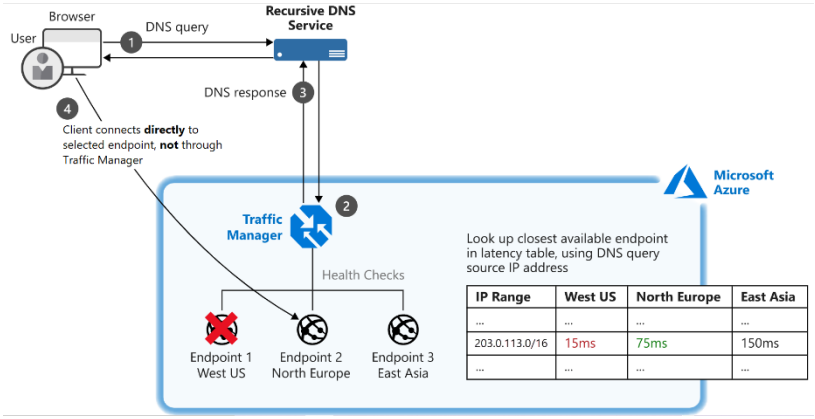
Weighted routing

Choose weighted when you want to distribute traffic across a set of endpoints, either evenly or based on different weights. The weight is an integer from 1 to 1,000. For each DNS query received, Traffic Manager randomly chooses an available endpoint. The probability of choosing an endpoint is based on the weights assigned to all available endpoints.



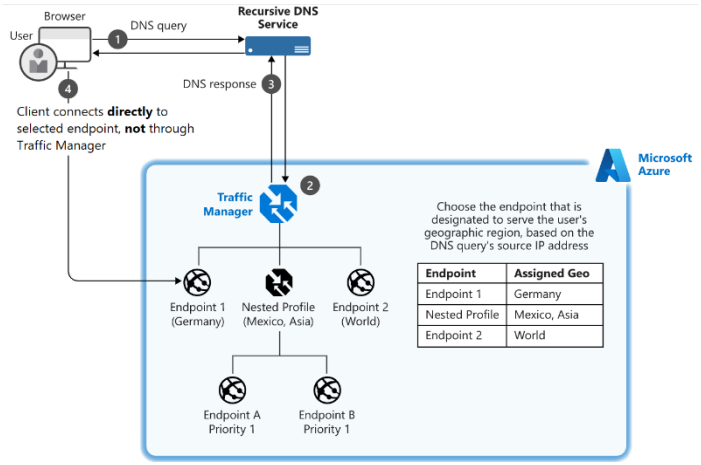
Performance routing

If you have endpoints in different geographic locations, you can use performance routing to send users to the endpoint that has the best performance for the user. To choose the best endpoint to use, this routing method uses an internet latency table, which actively tracks network latencies to the endpoints from locations around the globe. When a user makes a request, Traffic Manager returns the best-performing endpoint based on the location of the request.



Geographic routing

With the geographic routing method, users are directed to specific endpoints based on where their DNS query originates. Using this method allows you to geo-fence content to specific user regions. For example, European users can be directed to an endpoint in Europe that has specific terms and conditions for regional compliance. Users in China can be directed to an endpoint that's been localized in Mandarin.



Multivalue routing

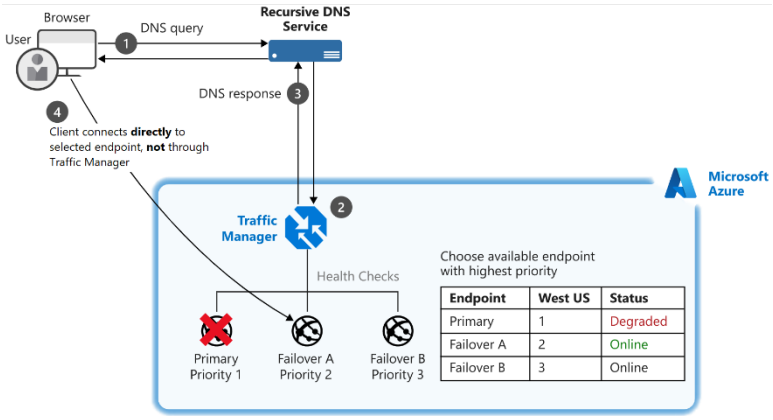
You can use the multivalue routing method to get multiple healthy endpoints in a single DNS query response. The caller can make client-side retries with other endpoints if an endpoint is unresponsive. This pattern can increase the availability of a service and reduce the latency associated with a new DNS query to obtain a healthy endpoint.

Subnet routing

This method maps the set of user IP address ranges to specific endpoints within a Traffic Manager profile. When Traffic Manager receives a request, the endpoint returned is the one mapped for that request's source IP address. For example, using subnet routing, a customer can route all requests from their corporate office to a different endpoint, where they might be testing an internal-only version of the app. Another scenario is if you want to provide a different experience to users who connect from a specific ISP (for example, to block users from a specific ISP).

Priority routing

The Traffic Manager profile contains a prioritized list of service endpoints. By default, Traffic Manager sends all traffic to the primary (highest-priority) endpoint. If the primary endpoint isn't available, Traffic Manager routes the traffic to the second endpoint. If both the primary and secondary endpoints aren't available, the traffic goes to the third endpoint, and so on. Availability of the endpoint is based on the configured status (enabled or disabled) and the ongoing endpoint monitoring that is set up.

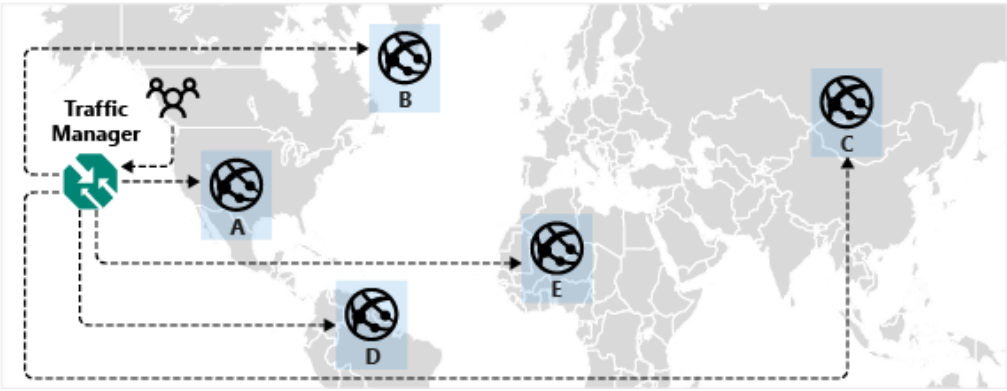


Optimize applications across regions by using performance routing

The *performance* traffic routing method connects users with the server that performs best for them. It might be better performing because it's physically closer to the user, but it might also be due to congestion or internet network connectivity. Azure stores historical DNS query latency for connecting clients in an internet latency table. Azure can use this information to direct traffic to the fastest-responding server, which is the server with the lowest latency. Traffic Manager maintains the internet latency table by tracking the roundtrip time between IP address ranges and each Azure datacenter. If an endpoint becomes unavailable, Traffic Manager doesn't include it in DNS query responses.

You don't have to do anything more than configure a Traffic Manager profile and select **Performance** as the routing method. Endpoints don't need to be prioritized; Traffic Manager will route all the traffic automatically to the fastest responding server.

In the following example, if endpoint A stopped performing as efficiently as endpoint B, customer traffic is automatically routed to endpoint B.



Client traffic is routed consistently. A client is directed to the same endpoint for each request it makes if nothing changes in the underlying servers and networking. If you need more granular control (for example, to choose a preferred failover within a region), you can use Traffic Manager in a nested configuration.

What is Azure Network Watcher?

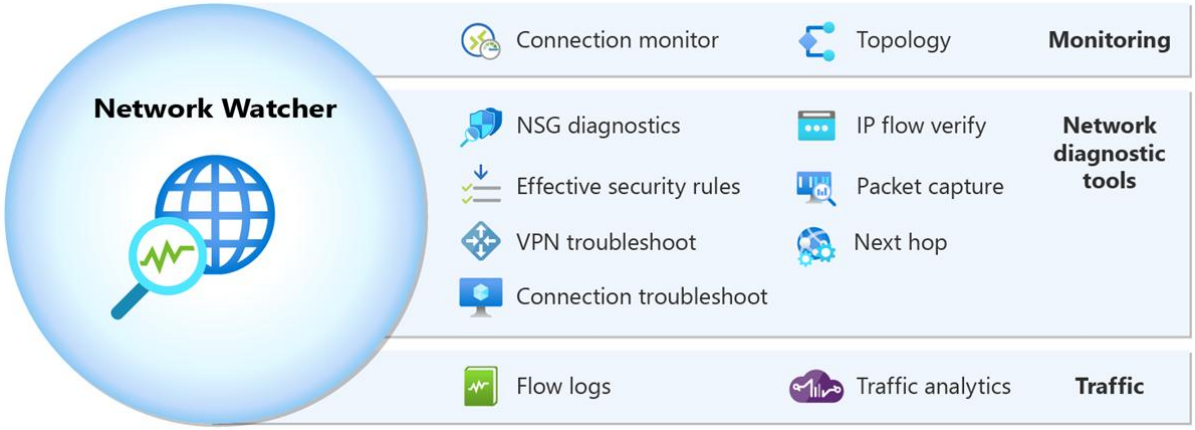
Azure Network Watcher provides a suite of tools to monitor, diagnose, view metrics, and enable or disable logs for Azure IaaS (Infrastructure-as-a-Service) resources. Network Watcher enables you to monitor and repair the network health of IaaS products like virtual machines (VMs), virtual networks (VNets), application gateways, load balancers, etc. Network Watcher isn't designed or intended for PaaS monitoring or Web analytics.

Network Watcher consists of three major sets of tools and capabilities:

- Monitoring
    - Topology
    - Connection monitor
- Network diagnostic
    - IP flow verify
    - NSG diagnostics
    - Next hop
    - Effective security rules
    - Connection troubleshoot
    - Packet capture

- VPN troubleshoot

- Traffic
  - Flow logs
  - Traffic analytics



Monitoring

Network Watcher offers two monitoring tools that help you view and monitor resources:

- Topology
- Connection monitor

Topology

The **Topology** tool provides a visualization of the entire network for understanding network configuration. It provides an interactive interface to view resources and their relationships in Azure spanning across multiple subscriptions, resource groups, and locations. At the beginning of the troubleshooting process, this tool helps you visualize all of the elements involved in the problem, allowing you to find something that isn't apparent by looking at the contents of resource groups.

Connection monitor

Connection monitor provides end-to-end connection monitoring for Azure and hybrid endpoints. It helps you understand network performance between various endpoints in your network infrastructure. You can use connection monitor to verify that two IaaS VMs that host the components of a multi-tier application can communicate with each other. You can also use it to verify connectivity in hybrid scenarios.

Network diagnostic tools

Network Watcher offers seven network diagnostic tools that help troubleshoot and diagnose network issues:

- IP flow verify
- NSG diagnostics
- Next hop
- Effective security rules
- Connection troubleshoot
- Packet capture
- VPN troubleshoot

IP flow verify

IP flow verify allows you to detect traffic filtering issues at a virtual machine level. It checks if a packet is allowed or denied to or from an IP address (IPv4 or IPv6 address). It also tells you which security rule allowed or denied the traffic.

NSG diagnostics

NSG diagnostics allows you to detect traffic filtering issues at a virtual machine, virtual machine scale set, or application gateway level. It checks if a packet is allowed or denied to or from an IP address, IP prefix, or a service tag. It tells you which security rule allowed or denied the traffic. It also allows you to add a new security rule with a higher priority to allow or deny the traffic.

Next hop

Next hop allows you to detect routing issues. It checks if traffic is routed correctly to the intended destination. It provides you with information about the Next hop type, IP address, and Route table ID for a specific destination IP address.

Effective security rules

Effective security rules allows you to view the effective security rules applied to a network interface. It shows you all security rules applied to the network interface, the subnet the network interface is in, and the aggregate of both.

Connection troubleshoot

Connection troubleshoot enables you to test a connection between a virtual machine, a virtual machine scale set, an application gateway, or a Bastion host and a virtual machine, an FQDN, a URI, or an IPv4 address. The test returns similar information returned when using the connection monitor tool, but tests the connection at a point in time instead of monitoring it over time, as connection monitor does.

Packet capture

Packet capture allows you to remotely create packet capture sessions to record all network traffic to and from a virtual machine (VM) or a virtual machine scale set.

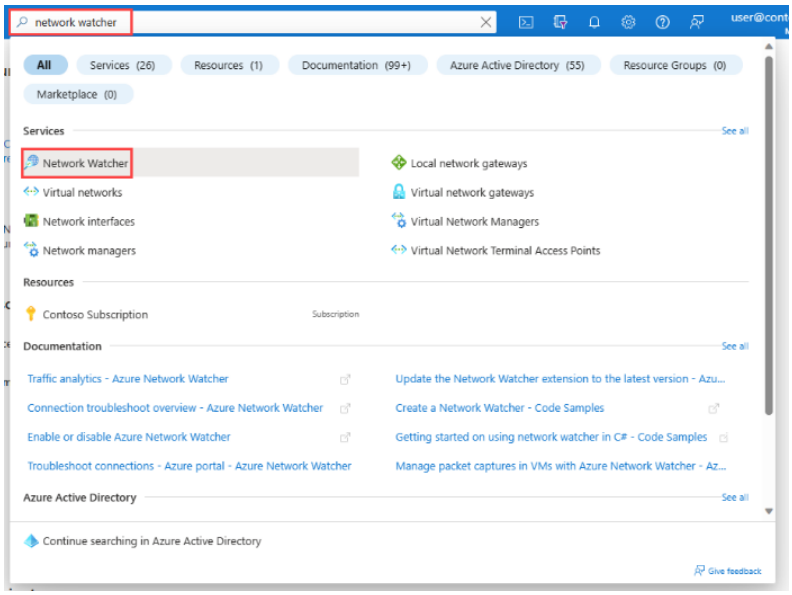
VPN troubleshoot

VPN troubleshoot enables you to troubleshoot virtual network gateways and their connections.

Traffic

Network Watcher offers two traffic tools that help you log and visualize network traffic:

- Flow logs
- Traffic analytics



Flow logs

- **Flow logs** allows you to log information about your Azure IP traffic and stores the data in Azure storage. You can log IP traffic flowing through a network security group or Azure virtual network.

Traffic analytics

- **Traffic analytics** provides rich visualizations of flow logs data.

How Azure Network Watcher works

Network Watcher becomes automatically available when you create a virtual network in an Azure region in your subscription. You can access Network Watcher directly in the Azure portal by typing **Network Watcher** in the **Search** bar.

Implement Azure Storage

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

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.

Things to know about Azure Storage

You can think of Azure Storage as supporting three categories of data: structured data, unstructured data, and virtual machine data. Review the following categories and think about which types of storage are used in your organization.

Category	Description	Storage examples
Virtual machine data	Virtual machine data storage includes disks and files. Disks are persistent block storage for Azure IaaS virtual machines. Files are fully managed file shares in the cloud.	Storage for virtual machine data is provided through Azure managed disks. Data disks are used by virtual machines to store data like database files, website static content, or custom application code. The number of data disks you can add depends on the virtual machine size. Each data disk has a maximum capacity of 32,767 GB.
Unstructured data	Unstructured data is the least organized. The format of unstructured data is referred to as *nonrelational*.	Unstructured data can be stored by using Azure Blob Storage and Azure Data Lake Storage. Blob Storage is a highly scalable, REST-based cloud object store. Azure Data Lake Storage is the Hadoop Distributed File System (HDFS) as a service.
Structured data	Structured data is stored in a relational format that has a shared schema. Structured data is often contained in a database table with rows, columns, and keys. Tables are an autoscaling NoSQL store.	Structured data can be stored by using Azure Table Storage, Azure Cosmos DB, and Azure SQL Database. Azure Cosmos DB is a globally distributed database service. Azure SQL Database is a fully managed database-as-a-service built on SQL.

Storage account types

General purpose Azure storage accounts have two types: Standard and Premium.

- **Standard** storage accounts are backed by magnetic hard disk drives (HDD). A standard storage account provides the lowest cost per GB. You can 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. You can use Premium storage for Azure virtual machine disks with I/O-intensive applications like databases.

Things to consider when using Azure Storage

As you think about your configuration plan for Azure Storage, consider these prominent features.

- **Consider durability and availability**. Azure Storage is durable and highly available. Redundancy ensures your data is safe during transient hardware failures. You replicate data across datacenters or geographical regions for protection from local catastrophe or natural disaster. Replicated data remains highly available during an unexpected outage.

- **Consider secure access**. Azure Storage encrypts all data. Azure Storage provides you with fine-grained control over who has access to your data.
- **Consider scalability**. Azure Storage is designed to be massively scalable to meet the data storage and performance needs of modern applications.
- **Consider manageability**. Microsoft Azure handles hardware maintenance, updates, and critical issues for you.
- **Consider data accessibility**. Data in Azure Storage is accessible from anywhere in the world over HTTP or HTTPS. Microsoft provides SDKs for Azure Storage in various languages. You can use .NET, Java, Node.js, Python, PHP, Ruby, Go, and the REST API. Azure Storage supports scripting in Azure PowerShell or the Azure CLI.

Explore Azure Storage services

Azure Storage offers four data services that can be accessed by using an Azure storage account:

- **Azure Blob Storage (containers)**: A massively scalable object store for text and binary data.
- **Azure Files**: Managed file shares for cloud or on-premises deployments.
- **Azure Queue Storage**: A messaging store for reliable messaging between application components.
- **Azure Table Storage**: A service that stores nonrelational structured data (also known as structured NoSQL data).

Azure Blob Storage

Azure Blob Storage is Microsoft's object storage solution for the cloud. Blob Storage is optimized for storing massive amounts of unstructured or *nonrelational* 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, the 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.

Note: You can access data from Azure Blob Storage **by using the NFS protocol**.

Azure Files

Azure Files enables you to set up highly available network file shares. Shares can be accessed by using the Server Message Block (SMB) protocol and the Network File System (NFS) protocol. Multiple virtual machines can share the same files with both read and write access. You can also read the files by using the REST interface or the storage client libraries.

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 virtual machines. 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. All users who have the share mounted should have full read/write access to the share.

Azure Queue Storage

Azure Queue Storage 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.

Consider a scenario where 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 is to use a queue. When the customer finishes the upload, you can write a message to the queue. Then you can use an Azure Function to retrieve the message from the queue and create the thumbnails. Each of the processing parts can be scaled separately, which gives you more control when tuning the configuration.

Azure Table Storage

Azure Table storage is a service that stores nonrelational structured data (also known as structured NoSQL data) in the cloud, providing a key/attribute store with a schemaless design. Because Table storage is schemaless, it's easy to adapt your data as the needs of your application evolve. Access to Table storage data is fast and cost-effective for many types of applications, and is typically lower in cost than traditional SQL for similar volumes of data. In addition to the existing Azure Table Storage service, there's a new Azure Cosmos DB Table API offering that provides throughput-optimized tables, global distribution, and automatic secondary indexes.

Things to consider when choosing Azure Storage services

As you think about your configuration plan for Azure Storage, consider the prominent features of the types of Azure Storage and which options support your application needs.

- **Consider storage optimization for massive data**. Azure Blob Storage is optimized for storing massive amounts of unstructured data. Objects in Blob Storage can be accessed from anywhere in the world via HTTP or HTTPS. Blob Storage is ideal for serving data directly to a browser, streaming data, and storing data for backup and restore.
- **Consider storage with high availability**. Azure Files supports highly available network file shares. On-premises apps use file shares for easy migration. By using Azure Files, all users can access shared data and tools. Storage account credentials provide file share authentication to ensure all users who have the file share mounted have the correct read/write access.
- **Consider storage for messages**. Use Azure Queue Storage to store large numbers of messages. Queue Storage is commonly used to create a backlog of work to process asynchronously.
- **Consider storage for structured data**. Azure Table Storage is ideal for storing structured, nonrelational data. It provides throughput-optimized tables, global distribution, and automatic secondary indexes. Because Azure Table Storage is part of Azure Cosmos DB, you have access to a fully managed NoSQL database service for modern app development.

Determine storage account types

Azure Storage offers several storage account options. Each storage account supports different features and has its own pricing model.

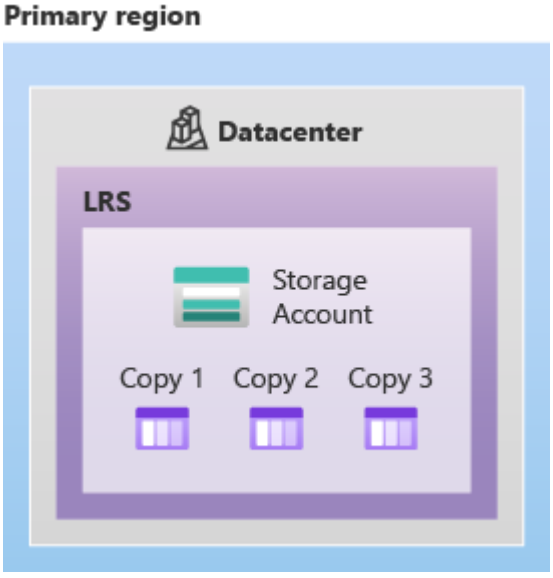
Things to know about storage account types

Review the following options and think about what storage accounts are required to support your applications.

Storage account	Supported services	Recommended usage
Standard general-purpose v2	Blob Storage (including Data Lake Storage), Queue Storage, Table Storage, and Azure Files	Standard storage account for most scenarios, including blobs, file shares, queues, tables, and disks (page blobs).
Premium block blobs	Blob Storage (including Data Lake Storage)	Premium storage account for block blobs and append blobs. Recommended for applications with high transaction rates. Use Premium block blobs if you work with smaller objects or require consistently low storage latency. This storage is designed to scale with your applications.
Premium file shares	Azure Files	Premium storage account for file shares only. Recommended for enterprise or high-performance scale applications. Use Premium file shares if you require support for both Server Message Block (SMB) and NFS file shares.
Premium page blobs	Page blobs only	Premium high-performance storage account for page blobs only. Page blobs are ideal for storing index-based and sparse data structures, such as operating systems, data disks for virtual machines, and databases.

Replication strategies

The data in your Azure storage account is always replicated to ensure durability and high availability. Azure Storage replication copies your data to protect from planned and unplanned events. These events range 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 your storage account meets the Service-Level Agreement (SLA) for Azure Storage even if there are failures.



We explore four replication strategies:

- Locally redundant storage (LRS)
- Zone redundant storage (ZRS)
- Geo-redundant storage (GRS)
- Geo-zone-redundant storage (GZRS)

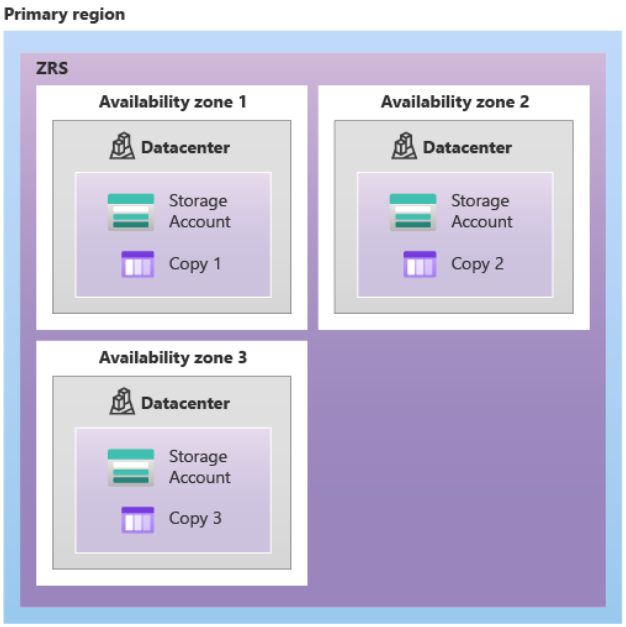
Locally redundant storage

Locally redundant storage is the lowest-cost replication option and offers the least durability compared to other strategies. If a data center-level disaster occurs, such as fire or flooding, all replicas might be lost or unrecoverable. Despite its limitations, LRS can be appropriate in several scenarios:

- Your application stores data that can be easily reconstructed if data loss occurs.
- Your data is constantly changing like in a live feed, and storing the data isn't essential.
- Your application is restricted to replicating data only within a country or region due to data governance requirements.

Zone redundant storage

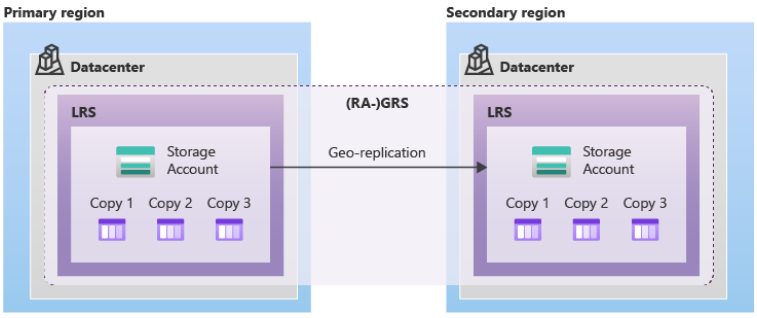
Zone redundant storage synchronously replicates your data across three storage clusters in a single region. Each storage cluster is physically separated from the others and resides in its own availability zone. Each availability zone, and the ZRS cluster within it, is autonomous, and has separate utilities and networking capabilities. Storing your data in a ZRS account ensures you can access and manage your data if a zone becomes unavailable. ZRS provides excellent performance and low latency.



- ZRS isn't currently 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.

Geo-redundant storage

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

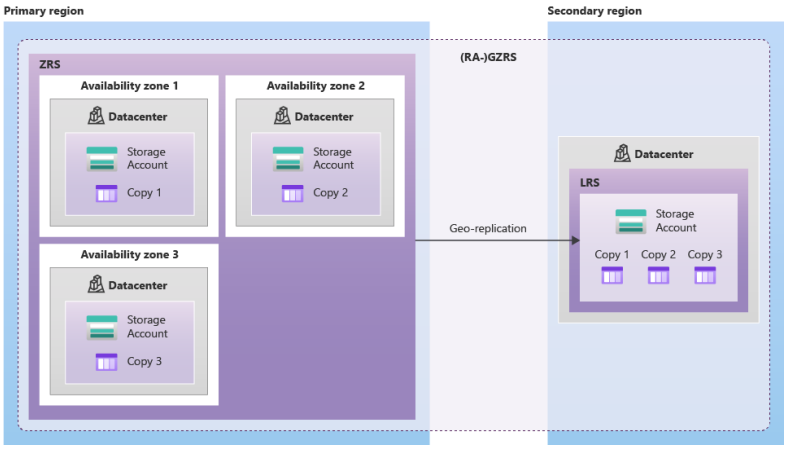


If you implement GRS, you have two related options to choose from:

- **GRS** replicates your data to another data center in a secondary region. The 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 region regardless of whether Microsoft initiates a failover from the primary to the secondary.

For a storage account with GRS or RA-GRS enabled, all data is first replicated with locally redundant storage. An update is first committed to the primary location and replicated by using LRS. The update is then replicated asynchronously to the secondary region by using GRS. Data in the secondary region uses 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.

Geo-zone redundant storage



Geo-zone-redundant storage 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 during a complete regional outage or during 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).

Access storage

Every object you store in Azure Storage has a unique URL address. Your storage account name forms the *subdomain* portion of the URL address. The combination of the subdomain and the domain name, which is specific to each service, forms an endpoint for your storage account.

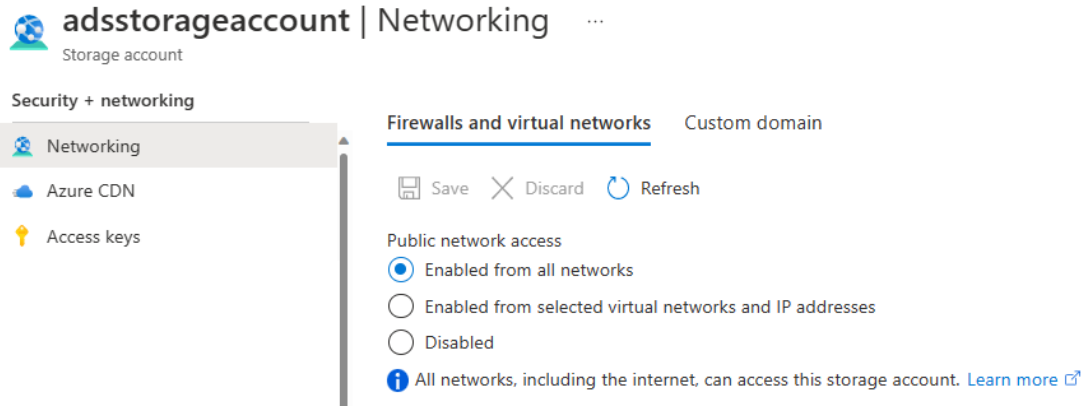
Service	Default endpoint
Container service	`// mystorageaccount .blob.core.windows.net`
Table service	`// mystorageaccount .table.core.windows.net`
Queue service	`// mystorageaccount .queue.core.windows.net`
File service	`// mystorageaccount .file.core.windows.net`

Let's look at an example. If your storage account name is *mystorageaccount*, default endpoints for your storage account are formed for the Azure services as shown in the following table:

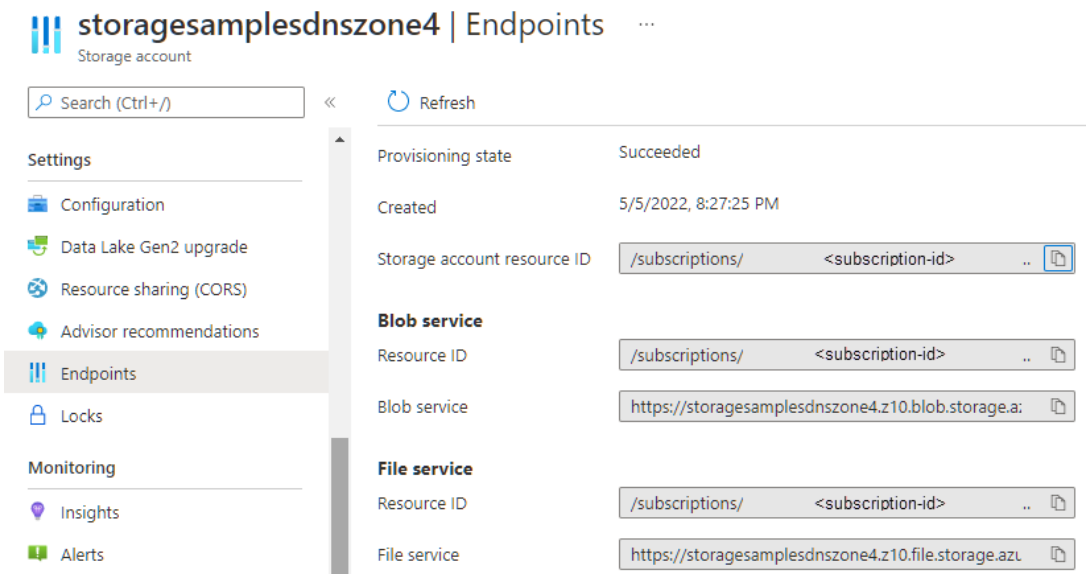
Secure storage endpoints

In the Azure portal, each Azure service requires certain steps to configure the service endpoints and restrict network access.

To access these settings for your storage account, you use the **Firewalls and virtual networks** settings. You add the virtual networks that should have access to the service for the account. - This setting restricts access to your storage account from specific subnets on virtual networks or public IPs.



The service endpoints for a storage account provide the base URL for any blob, queue, table, or file object in Azure Storage. Use this base URL to construct the address for any given resource.



Things to know about configuring service endpoints

Here are some points to consider about configuring service access settings:

- You can configure the service to allow access to one or more public IP ranges.
- Subnets and virtual networks must exist in the same Azure region or region pair as your storage account.

Implement Azure Blob Storage

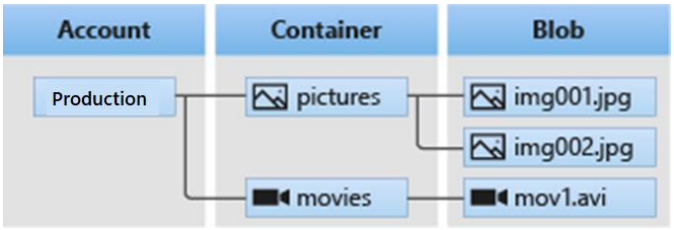
Azure Blob Storage is a service that stores unstructured data in the cloud as objects or blobs. Blob stands for Binary Large Object. Blob Storage is also referred to as *object storage* or *container storage*.

Things to know about Azure Blob Storage

Let's examine some configuration characteristics of Blob Storage.

- Blob Storage can store any type of text or binary data. Some examples are text documents, images, video files, and application installers.
- Blob Storage uses three resources to store and manage your data:
    - An Azure storage account
    - Containers in an Azure storage account
    - Blobs in a container
- To implement Blob Storage, you configure several settings:
    - Blob container options.
    - Blob types and upload options.
    - Blob Storage access tiers.
    - Blob lifecycle rules.
    - Blob object replication options.

The following diagram shows the relationship between the Blob Storage resources.



Things to consider when implementing Azure Blob Storage

- **Consider browser uploads**. Use Blob Storage to serve images or documents directly to a browser.
- **Consider distributed access**. Blob Storage can store files for distributed access, such as during an installation process.
- **Consider streaming data**. Stream video and audio by using Blob Storage.
- **Consider archiving and recovery**. Blob Storage is a great solution for storing data for backup and restore, disaster recovery, and archiving.
- **Consider application access**. You can store data in Blob Storage for analysis by an on-premises or Azure-hosted service.

Create blob containers

Azure Blob Storage uses a container resource to group a set of blobs. A blob can't exist by itself in Blob Storage. A blob must be stored in a container resource.

Things to know about containers and blobs

Let's look at the configuration characteristics of containers and blobs.

- All blobs must be in a container.
- A container can store an unlimited number of blobs.
- An Azure storage account can contain an unlimited number of containers.
- You can create the container in the Azure portal.
- You can upload blobs into a container.

Configure a container

In the Azure portal, you configure two settings to create a container for an Azure storage account. As you review these details, consider how you might organize containers in your storage account.

- **Name**: Enter a name for your container. The name must be unique within the Azure storage account.
    - The name can contain only lowercase letters, numbers, and hyphens.
    - The name must begin with a letter or a number.
    - The minimum length for the name is three characters.
    - The maximum length for the name is 63 characters.
- **Public access level**: The access level specifies whether the container and its blobs can be accessed publicly. By default, container data is private and visible only to the account owner. There are three access level choices:
    - **Private**: (Default) Prohibit anonymous access to the container and blobs.
    - **Blob**: Allow anonymous public read access for the blobs only.
    - **Container**: Allow anonymous public read and list access to the entire container, including the blobs.

Note: You can also create a blob container with PowerShell by using the New-AzStorageContainer command.

Assign blob access tiers

Azure Storage supports several access tiers for blob data. These tiers include Hot, Cool, Cold, and Archive. Each access tier is optimized to support a particular pattern of data usage.

Things to know about blob access tiers

Let's examine characteristics of the blob access tiers.

Hot tier

The Hot tier is optimized for frequent reads and writes of objects in the Azure storage account. A good usage case is data that is actively being processed. An online tier optimized for storing data that is accessed or modified frequently. The hot tier has the highest storage costs, but the lowest access costs.

Cool tier

The Cool tier is optimized for storing large amounts of infrequently accessed data. This tier is intended for data that remains in the Cool tier for at least 30 days. A usage case for the Cool tier is short-term backup and disaster recovery datasets and older media content. This content shouldn't be viewed frequently, but it needs to be immediately

available. Storing data in the Cool tier is more cost-effective. The cool tier has lower storage costs and higher access costs compared to the hot tier.

Cold tier

The Cold tier is also optimized for storing large amounts of infrequently accessed data. This tier is intended for data that can remain in the tier for at least 90 days. The cold tier has lower storage costs and higher access costs compared to the cool tier.

Archive tier

The Archive tier is an offline tier that's optimized for data that can tolerate several hours of retrieval latency. Data must remain in the Archive tier for at least 180 days or be subject to an early deletion charge. Data for the Archive tier includes secondary backups, original raw data, and legally required compliance information. This tier is the most cost-effective option for storing data. Accessing data is more expensive in the Archive tier than accessing data in the other tiers.

Compare access tiers

The access options for Azure Blob Storage offer a range of features and support levels to help you optimize your storage costs. As you compare the features and support, think about which access options can best support your application needs.

Comparison	Hot access tier	Cool access tier	Cold access tier	Archive access tier
Availability	99.9%	99%	99%	99%
Availability (RA-GRS reads)	99.99%	99.9%	99.9%	99.9%
Latency (time to first byte)	milliseconds	milliseconds	milliseconds	hours
Minimum storage duration	N/A	30 days	90 days	180 days

Add blob lifecycle management rules

Every data set has a unique lifecycle. Early in the lifecycle, users tend to access some of the data in the set, but not all the data. As the data set ages, access to all the data in the set tends to dramatically reduce. Some data set stays idle in the cloud and is rarely accessed. Some data expires within a few days or months after creation. Other data is actively read and modified throughout the data set lifetime.

Azure Blob Storage supports lifecycle management for data sets. It offers a rich rule-based policy for GPv2 and Blob Storage accounts. You can use lifecycle policy rules to transition your data to the appropriate access tiers and set expiration times for the end of a data set's lifecycle.

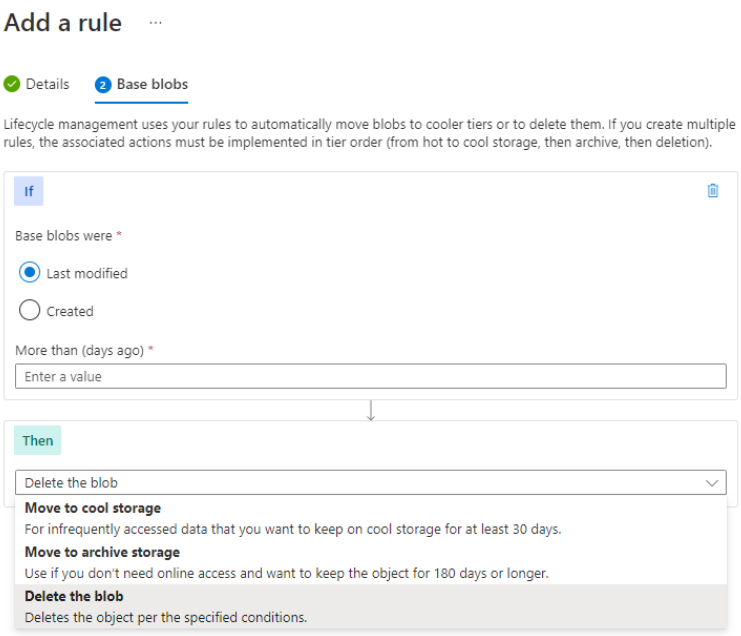
Things to know about lifecycle management

You can use Azure Blob Storage lifecycle management policy rules to accomplish several tasks.

- Transition blobs to a cooler storage tier (Hot to Cool, Hot to Archive, Cool to Archive) to optimize for performance and cost.
- Delete blobs at the end of their lifecycles.
- Define rule-based conditions to run once per day at the Azure storage account level.
- Apply rule-based conditions to containers or a subset of blobs.

Configure lifecycle management policy rules

In the Azure portal, you create lifecycle management policy rules for your Azure storage account by specifying several settings. For each rule, you create **If - Then** block conditions to transition or expire data based on your specifications. As you review these details, consider how you can set up lifecycle management policy rules for your data sets.



- **If**: The **If** clause sets the evaluation clause for the policy rule. When the **If** clause evaluates to true, the **Then** clause is executed. Use the **If** clause to set the time period to apply to the blob data. The lifecycle management feature checks if the data is accessed or modified according to the specified time.
    - **More than (days ago)**: The number of days to use in the evaluation condition.
- **Then**: The **Then** clause sets the action clause for the policy rule. When the **If** clause evaluates to true, the **Then** clause is executed. Use the **Then** clause to set the transition action for the blob data. The lifecycle management feature transitions the data based on the setting.
    - **Move to cool storage**: The blob data is transitioned to Cool tier storage.
    - **Move to cold storage**: The blob data is transitioned to Cold tier storage.
    - **Move to archive storage**: The blob data is transitioned to Archive tier storage.
    - **Delete the blob**: The blob data is deleted.

Upload blobs

A blob can be any type of data and any size file. Azure Storage offers three types of blobs: *block blob*, *page blob*, and *append blob*.

Things to know about blob types

Let's take a closer look at the characteristics of blob types.

- **Block blobs**. A block blob consists of blocks of data that are assembled to make a blob. Most Blob Storage scenarios use block blobs. Block blobs are ideal for storing text and binary data in the cloud, like files, images, and videos.
- **Append blobs**. An append blob is similar to a block blob because the append blob also consists of blocks of data. The blocks of data in an append blob are optimized for *append* operations. Append blobs are useful for logging scenarios, where the amount of data can increase as the logging operation continues.
- **Page blobs**. A page blob can be up to 8 TB in size. Page blobs are more efficient for frequent read/write operations. Azure Virtual Machines uses page blobs for operating system disks and data disks.
- The block blob type is the default type for a new blob. When you're creating a new blob, if you don't choose a specific type, the new blob is created as a block blob.
- After you create a blob, you can't change its type.

Things to consider when using blob upload tools

Upload tool	Description
AzCopy	An easy-to-use command-line tool for Windows and Linux. You can copy data to and from Blob Storage, across containers, and across storage accounts.
Azure Data Box Disk	A service for transferring on-premises data to Blob Storage when large datasets or network constraints make uploading data over the wire unrealistic. You can use Azure Data Box Disk to request solid-state disks (SSDs) from Microsoft. You can copy your data to those disks and ship them back to Microsoft to be uploaded into Blob Storage.
Azure Import/Export	A service that helps you export large amounts of data from your storage account to hard drives that you provide and that Microsoft then ships back to you with your data.

Determine Blob Storage pricing

All Azure storage accounts use a pricing model for Azure Blob Storage tiers. Total cost of block blob storage depends on:

- Volume of data stored per month.
- Quantity and types of operations performed, along with any data transfer costs.
- Data redundancy option selected.

Things to know about pricing for Blob Storage

Review the following billing considerations for an Azure storage account and Blob Storage.

- **Performance tiers**. The Blob Storage tier determines the amount of data stored and the cost for storing that data. As the performance tier gets cooler, the per-gigabyte cost decreases.
- **Data access costs**. Data access charges increase as the tier gets cooler. For data in the Cool and Archive tiers, you're billed a per-gigabyte data access charge for reads.
- **Transaction costs**. There's a per-transaction charge for all tiers. The charge increases as the tier gets cooler.
- **Geo-replication data transfer costs**. This charge only applies to accounts that have geo-replication configured, including GRS and RA-GRS. Geo-replication data transfer incurs a per-gigabyte charge.
- **Outbound data transfer costs**. Outbound data transfers incur billing for bandwidth usage on a per-gigabyte basis. This billing is consistent with general-purpose Azure storage accounts.
- **Changes to the storage tier**. If you change the account storage tier from Cool to Hot, you incur a charge equal to reading all the data existing in the storage account. Changing the account storage tier from Hot to Cool incurs a charge equal to writing all the data into the Cool tier (GPv2 accounts only).

Azure Storage security strategies

Administrators use different strategies to ensure their data is secure. Common approaches include encryption, authentication, authorization, and user access control with credentials, file permissions, and private signatures. Azure Storage offers a suite of security capabilities based on common strategies to help you secure your data.

Things to know about Azure Storage security strategies

Let's look at some characteristics of Azure Storage security.

- **Encryption at rest**. Storage Service Encryption (SSE) with a 256-bit Advanced Encryption Standard (AES) cipher encrypts all data written to Azure Storage. When you read data from Azure Storage, Azure Storage decrypts the data before returning it. This process incurs no extra charges and doesn't degrade performance. It can't be disabled.

- **Authentication**. Microsoft Entra ID and role-based access control (RBAC) are supported for Azure Storage for both resource management operations and data operations.
    - Assign RBAC roles scoped to an Azure storage account to security principals, and use Microsoft Entra ID to authorize resource management operations like key management.
    - Microsoft Entra integration is supported for data operations on Azure Blob Storage and Azure Queue Storage.
- **Encryption in transit**. Keep your data secure by enabling transport-level security between Azure and the client. Always use HTTPS to secure communication over the public internet. When you call the REST APIs to access objects in storage accounts, you can enforce the use of HTTPS by requiring secure transfer for the storage account. After you enable secure transfer, connections that use HTTP will be refused. This flag will also enforce secure transfer over SMB by requiring SMB 3.0 for all file share mounts.
- **Disk encryption**. For virtual machines (VMs), Azure lets you encrypt virtual hard disks (VHDs) by using Azure Disk Encryption. This encryption uses BitLocker for Windows images, and uses dm-crypt for Linux. Azure Key Vault stores the keys automatically to help you control and manage the disk-encryption keys and secrets. So even if someone gets access to the VHD image and downloads it, they can't access the data on the VHD.
- **Shared access signatures**. Delegated access to the data objects in Azure Storage can be granted by using a shared access signature (SAS).
- **Authorization**. Every request made against a secured resource in Blob Storage, Azure Files, Queue Storage, or Azure Cosmos DB (Azure Table Storage) must be authorized. Authorization ensures that resources in your storage account are accessible only when you want them to be, and to only those users or applications whom you grant access.

Things to consider when using authorization security

Review the following strategies for authorizing requests to Azure Storage. Think about what security strategies would work for your Azure Storage.

Authorization strategy	Description
Microsoft Entra ID	Microsoft Entra ID is Microsoft's cloud-based identity and access management service. With Microsoft Entra ID, you can assign fine-grained access to users, groups, or applications by using role-based access control.
Shared Key	Shared Key authorization relies on your Azure storage account access keys and other parameters to produce an encrypted signature string. The string is passed on the request in the Authorization header.
Shared access signatures	A SAS delegates access to a particular resource in your Azure storage account with specified permissions and for 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 don't require authorization.

Shared Access Signatures

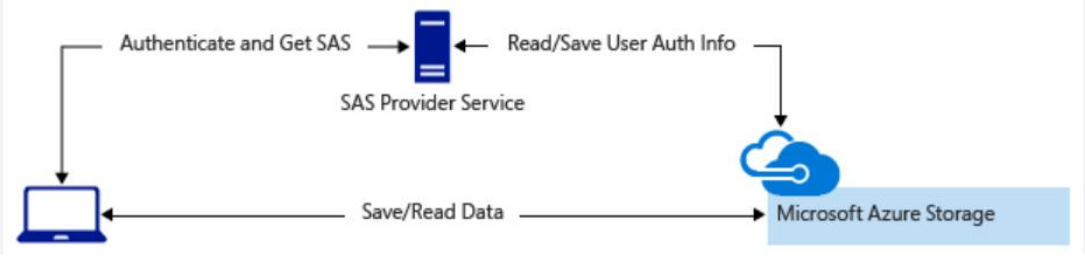
A shared access signature (SAS) is a uniform resource identifier (URI) that grants restricted access rights to Azure Storage resources. SAS is a secure way to share your storage resources without compromising your account keys.

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.

You'd typically use a SAS for a service where users read and write their data to your storage account. Accounts that store user data have two typical designs:

- Clients can upload and download data through a front-end proxy service, which performs authentication. This front-end proxy service has the advantage of allowing validation of business rules. if you handle large amounts of data or high-volume transactions, it can be difficult to scale this service

- A lightweight service authenticates the client, as needed. Next, it generates a SAS. Clients receiving the SAS can access storage account resources directly. The SAS defines the client's permissions and access interval. It reduces the need to route all data through the front-end proxy service.



Things to know about shared access signatures

Let's review some characteristics of a SAS.

- A SAS gives you granular control over the type of access you grant to clients who have the SAS.
- An account-level SAS can delegate access to multiple Azure Storage services, such as blobs, files, queues, and tables.
- You can specify the time interval for which a SAS is valid, including the start time and the expiration time.
- You specify the permissions granted by the SAS. A SAS for a blob might grant read and write permissions to that blob, but not delete permissions.
- SAS provides account-level and service-level control.
    - **Account-level**. Use an *account-level SAS* to allow access to anything that a service-level SAS can allow, plus other resources and abilities. For example, you can use an account-level SAS to allow the ability to create file systems.
    - **Service-level**.You can use a *service-level SAS* to allow access to specific resources in a storage account. You'd use this type of SAS, for example, to allow an app to retrieve a list of files in a file system, or to download a file.

Note: A **stored access policy** can provide another level of control when you use a service-level SAS on the server side.

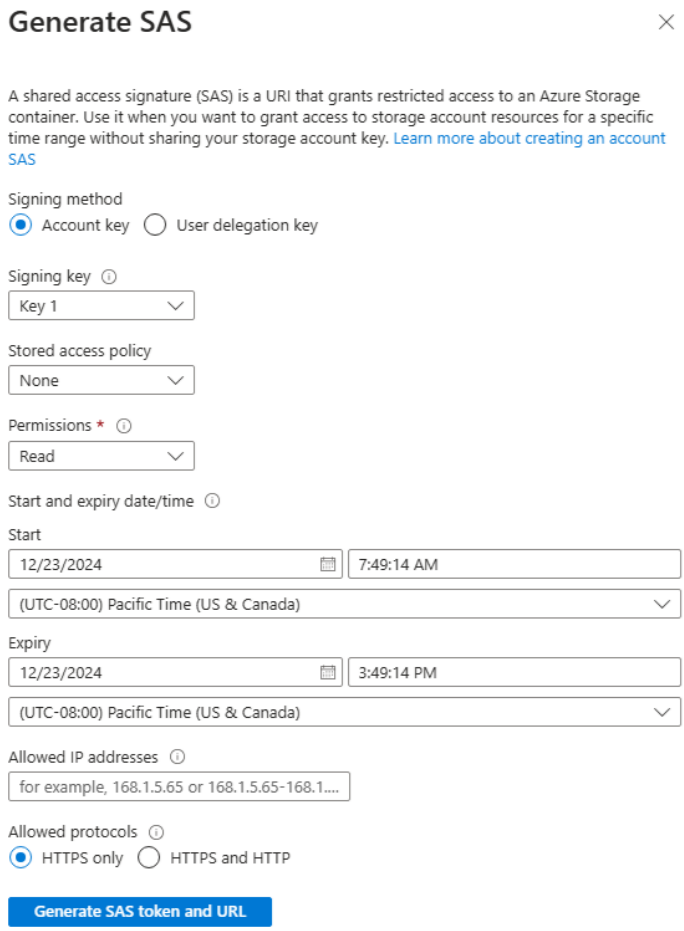
You can group SASs and provide other restrictions by using a stored access policy.

- There are optional SAS configuration settings:
    - **IP addresses**. You can identify an IP address or range of IP addresses from which Azure Storage accepts the SAS. Configure this option to specify a range of IP addresses that belong to your organization.
    - **Protocols**. You can specify the protocol over which Azure Storage accepts the SAS. Configure this option to restrict access to clients by using HTTPS.

Configure a shared access signature

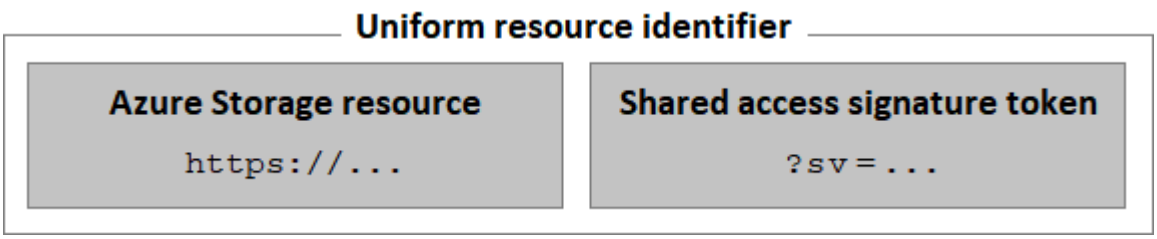
In the Azure portal, you configure several settings to create a SAS. As you review these details, consider how you might implement shared access signatures in your storage security solution.

- **Signing method**: Choose the signing method: Account key or User delegation key.
- **Signing key**: Select the signing key from your list of keys.
- **Permissions**: Select the permissions granted by the SAS, such as read or write.
- **Start and Expiry date/time**: Specify the time interval for which the SAS is valid. Set the start time and the expiry time.
- **Allowed IP addresses**: (Optional) Identify an IP address or range of IP addresses from which Azure Storage accepts the SAS.
- **Allowed protocols**: (Optional) Select the protocol over which Azure Storage accepts the SAS.



Identify URI and SAS parameters

When you create your shared access signature (SAS), a uniform resource identifier (URI) is created by using parameters and tokens. The URI consists of your Azure Storage resource URI and the SAS token.



Things to know about URI definitions

Let's look at an example URI definition and examine the parameters. This example creates a service-level SAS that grants read and write permissions to a blob. Consider how you might configure the parameters to support your Azure Storage resources.

https://myaccount.blob.core.windows.net/?restype=service&comp=properties&sv=2015-04-05&ss=bf&st=2015-04-29T22%3A18%3A26Z&se=2015-04-30T02%3A23%3A26Z&sr=b&sp=rw&sip=168.1.5.60-168.1.5.70&spr=https&sig=F%6GRVAZ5Cdj2Pw4tgU7IlSTkWgn7bUkkAg8P6HESXwmf%4B

Parameter	Example	Description
Resource URI	`https://myaccount.blob.core.windows.net/?restype=service&comp=properties`	Defines the Azure Storage endpoint and other parameters. This example defines an endpoint for Blob Storage and indicates that the SAS applies to service-level operations. When the URI is used with `GET`, the Storage properties are retrieved. When the URI is used with `SET`, the Storage properties are configured.
Storage version	`sv=2015-04-05`	For Azure Storage version 2012-02-12 and later, this parameter indicates the version to use. This example indicates that version 2015-04-05 (April 5, 2015) should be used.
Storage service	`ss=bf`	Specifies the Azure Storage to which the SAS applies. This example indicates that the SAS applies to Blob Storage and Azure Files.
Start time	`st=2015-04-29T22%3A18%3A26Z`	(Optional) Specifies the start time for the SAS in UTC time. This example sets the start time as April 29, 2015 22:18:26 UTC. If you want the SAS to be valid immediately, omit the start time.
Expiry time	`se=2015-04-30T02%3A23%3A26Z`	Specifies the expiration time for the SAS in UTC time. This example sets the expiry time as April 30, 2015 02:23:26 UTC.
Resource	`sr=b`	Specifies which resources are accessible via the SAS. This example specifies that the accessible resource is in Blob Storage.
Permissions	`sp=rw`	Lists the permissions to grant. This example grants access to read and write operations.
IP range	`sip=168.1.5.60-168.1.5.70`	Specifies a range of IP addresses from which a request is accepted. This example defines the IP address range 168.1.5.60 through 168.1.5.70.
Protocol	`spr=https`	Specifies the protocols from which Azure Storage accepts the SAS. This example indicates that only requests by using HTTPS are accepted.
Signature	`sig=F%6GRVAZ5Cdj2Pw4tgU7Il STkWgn7bUkkAg8P6HESXwmf%4B`	Specifies that access to the resource is authenticated by using a Hash-Based Message Authentication Code (HMAC) signature. The signature is computed with a key using the SHA256 algorithm, and encoded by using Base64 encoding.

Azure Storage encryption

Azure Storage encryption for data at rest protects your data by ensuring your organizational security and compliance commitments are met. The encryption and decryption processes happen automatically. Because your data is secured by default, you don't need to modify your code or applications.

When you create a storage account, Azure generates two 512-bit storage account access keys for that account. These keys can be used to authorize access to data in your storage account via Shared Key authorization, or via SAS tokens that are signed with the shared key.

Microsoft recommends that you use Azure Key Vault to manage your access keys, and that you regularly rotate and regenerate your keys. Using Azure Key Vault makes it easy to rotate your keys without interruption to your applications. You can also manually rotate your keys.

Things to know about Azure Storage encryption

Examine the following characteristics of Azure Storage encryption.

- Data is automatically encrypted before written to Azure storage.
- Data is automatically decrypted when retrieved.
- Azure Storage encryption, encryption at rest, decryption, and key management are transparent to users.
- All data written to Azure Storage is encrypted through 256-bit advanced encryption standard (AES) encryption. AES is one of the strongest block ciphers available.
- Azure Storage encryption is enabled for all new and existing storage accounts and can't be disabled.

Configure Azure Storage encryption

In the Azure portal, you configure Azure Storage encryption by specifying the encryption type. You can manage the keys yourself, or you can have the keys managed by Microsoft. Consider how you might implement Azure Storage encryption for your storage security.

- **Infrastructure encryption**. Infrastructure encryption can be enabled for the entire storage account, or for an encryption scope within an account. When infrastructure encryption is enabled for a storage account or an encryption scope, data is encrypted twice—once at the service level and once at the infrastructure level—with two different encryption algorithms and two different keys.
- **Platform-managed keys**. Platform-managed keys (PMKs) are encryption keys generated, stored, and managed entirely by Azure. Customers don't interact with PMKs. The keys used for Azure Data Encryption-at-Rest, for instance, are PMKs by default.
- **Customer-managed keys**. Customer managed keys (CMK), on the other hand, are keys read, created, deleted, updated, and/or administered by one or more customers. Keys stored in a customer-owned key vault or hardware security module (HSM) are CMKs. Bring Your Own Key (BYOK) is a CMK scenario in which a customer imports (brings) keys from an outside storage location.

Customer-Managed Keys (CMK)

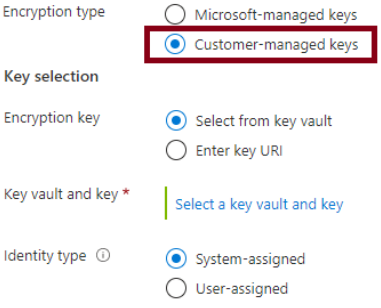
For your Azure Storage security solution, you can use Azure Key Vault to manage your encryption keys. The Azure Key Vault APIs can be used to generate encryption keys. You can also create your own encryption keys and store them in a key vault.

Things to know about customer-managed keys

Consider the following characteristics of customer-managed keys.

- By creating your own keys (referred to as *customer-managed* keys), you have more flexibility and greater control.
- You can create, disable, audit, rotate, and define access controls for your encryption keys.
- Customer-managed keys can be used with Azure Storage encryption. You can use a new key or an existing key vault and key. The Azure storage account and the key vault must be in the same region, but they can be in different subscriptions.

Configure customer-managed keys



In the Azure portal, you can configure customer-managed encryption keys. You can create your own keys, or you can have the keys managed by Microsoft. Consider how you might use Azure Key Vault to create your own customer-managed encryption keys.

- **Encryption type**: Choose how the encryption key is managed: by Microsoft or by yourself (customer).
- **Encryption key**: Specify an encryption key by entering a URI or select a key from an existing key vault.

Configure Azure Files

Azure Files offers fully managed file shares in the cloud. You can access Azure file shares by using the Server Message Block (SMB), Network File System (NFS), and HTTP protocols. Clients can connect to Azure file shares from Windows, Linux, and macOS devices.

Things to know about Azure Files

Here are some characteristics of Azure files:

- **Serverless deployment**. An Azure file share is a PaaS offering of a fully managed file share that doesn't require any infrastructure. You don't need to take care of any VMs, operating systems, or updates.
- **Almost unlimited storage**. A single Azure file share can store up to 100 tebibytes (TiB) of files, and a file can be up to 4 TiB in size. The files are organized in a hierarchical folder structure in the same way as on on-premises file servers.
- **Data encryption**. The data on an Azure file share is encrypted at rest in an Azure datacenter and in transit on a network.
- **Access from anywhere**. By default, clients can access Azure file shares from anywhere if they have internet connectivity.
- **Integration into an existing environment**. You can control access to Azure file shares by using Microsoft Entra identities or AD DS identities that are synced to Microsoft Entra ID. This helps ensure that users can have the same experience accessing an Azure file share as when they access an on-premises file server.
- **Previous versions and backups**. You can create Azure file share snapshots that integrate with the Previous Versions feature in File Explorer. You can also use Azure Backup to back up Azure file shares.
- **Data redundancy**. Azure file share data replicates to multiple locations in the same Azure datacenter or across many Azure datacenters. The replication setting of the Azure storage account that includes the file share controls the data redundancy.

Things to consider when using Azure Files

There are many common scenarios for using Azure Files. As you review the following suggestions, think about how Azure Files can provide solutions for your organization.

- **Consider replacement and supplement options**. Replace or supplement traditional on-premises file servers or NAS devices by using Azure Files.
- **Consider global access**. Directly access Azure file shares by using most operating systems, such as Windows, macOS, and Linux, from anywhere in the world.
- **Consider lift and shift support**. *Lift and shift* applications to the cloud with Azure Files for apps that expect a file share to store file application or user data.
- **Consider using Azure File Sync**. Replicate Azure file shares to Windows Servers by using Azure File Sync. You can replicate on-premises or in the cloud for performance and distributed caching of the data where it's being used. We take a closer look at Azure File Sync in a later unit.
- **Consider shared applications**. Store shared application settings such as configuration files in Azure Files.
- **Consider diagnostic data**. Use Azure Files to store diagnostic data such as logs, metrics, and crash dumps in a shared location.
- **Consider tools and utilities**. Azure Files is a good option for storing tools and utilities that are needed for developing or administering Azure VMs or cloud services.

Compare Azure Files to Azure Blob Storage

It's important to understand when to use Azure Files to store data in file shares rather than using Azure Blob Storage to store data as blobs. The following table compares different features of these services and common implementation scenarios.

Azure Files (file shares)	Azure Blob Storage (blobs)
Azure Files provides the SMB and NFS protocols, client libraries, and a REST interface that allows access from anywhere to stored files.	Azure Blob Storage provides client libraries and a REST interface that allows unstructured data to be stored and accessed at a massive scale in block blobs.
- Files in an Azure Files share are true directory objects. - Data in Azure Files is accessed through file shares across multiple virtual machines.	- Blobs in Azure Blob Storage are a flat namespace. - Blob data in Azure Blob Storage is accessed through a container.
Azure Files *is ideal to lift and shift an application to the cloud that already uses the native file system APIs. Share data between the app and other applications running in Azure.* *Azure Files is a good option when you want to store development and debugging tools that need to be accessed from many virtual machines.*	***Azure Blob Storage*** *is ideal for applications that need to support streaming and random-access scenarios.* *Azure Blob Storage is a good option when you want to be able to access application data from anywhere.*

Manage Azure file shares

Azure Files offers two industry-standard file system protocols for mounting Azure file shares: the Server Message Block (SMB) protocol and the Network File System (NFS) protocol. Azure file shares don't support both the SMB and NFS protocols on the same file share, although you can create SMB and NFS Azure file shares within the same storage account.

Types of Azure file shares

Azure Files supports two storage tiers: premium and standard. Standard file shares are created in general purpose (GPv2) storage accounts, while premium file shares are created in FileStorage storage accounts. The two storage tiers have the attributes described in the following table.

Storage tier	Description
Premium	Premium file shares store data on solid-state drives (SSDs), and are available only in the FileStorage storage account kind. They provide consistent high performance and low latency, and are available in LRS redundancy, with ZRS available in some regions. Not available in all Azure regions.
Standard	Standard file shares store data on hard disk drives (HDDs) and deploy in the general-purpose version 2 (GPv2) storage account type. Provide performance for workloads such as general-purpose file shares and dev/test environments. Standard file shares are available for LRS, ZRS, GRS, and GZRS, in all Azure regions.

Types of authentications

Three are three main authentications methods that Azure Files supports.

Authentication method	Description
Identity-based authentication over SMB	Provides the same seamless single sign-on (SSO) experience when accessing Azure file shares as accessing on-premises file shares.
Access key	An access key is an older and less flexible option. An Azure storage account has two access keys that can be used when making a request to the storage account, including to Azure Files. Access keys are static and provide full control access to Azure Files. Access keys should be secured and not shared with users, because they bypass all access control restrictions. A best practice is to avoid sharing storage account keys and use identity-based authentication whenever possible.
A Shared Access Signature (SAS) token	SAS is a dynamically generated Uniform Resource Identifier (URI) that's based on the storage access key. SAS provides restricted access rights to an Azure storage account. Restrictions include allowed permissions, start and expiry time, allowed IP addresses from where requests can be sent, and allowed protocols. With Azure Files, a SAS token is only used to provide REST API access from code.

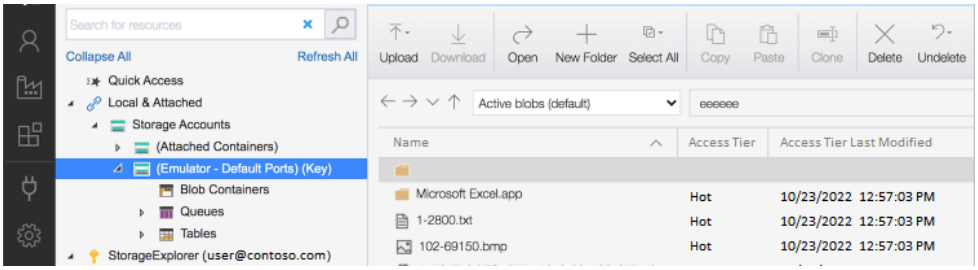
Creating SMB Azure file shares

There are two important settings that you need to be aware of when creating and configuring SMB Azure file shares.

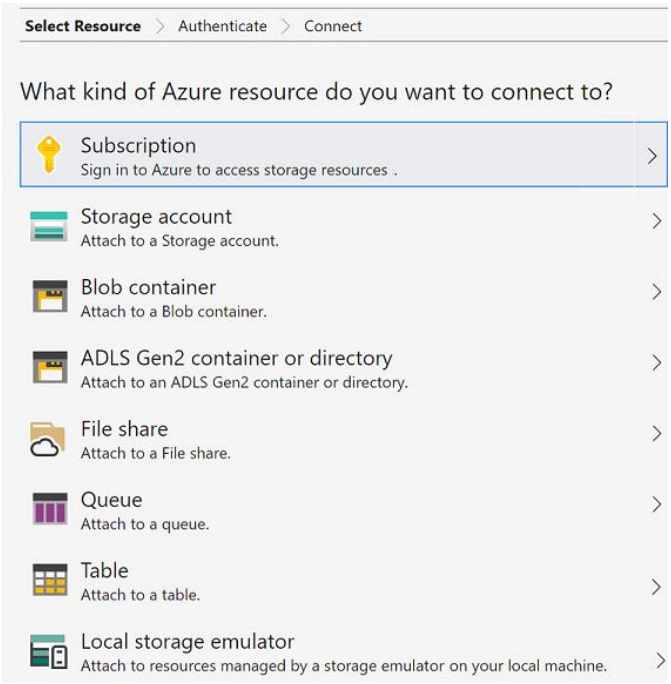
- **Open port 445**. SMB communicates over TCP port 445. Make sure port 445 is open. Also, make sure your firewall isn't blocking TCP port 445 from the client machine. If you can't unblock port 445, then a VPN or ExpressRoute connection from on-premises to your Azure network is required, with Azure Files exposed on your internal network using private endpoints.
- **Enable secure transfer**. The Secure transfer required setting enhances the security of your storage account by limiting requests to your storage account from secure connections only. Consider the scenario where you use REST APIs to access your storage account. If you attempt to connect, and secure transfer required is enabled, you must connect by using HTTPS. If you try to connect to your account by using HTTP, and secure transfer required is enabled, the connection is rejected.

Azure Storage Explorer

Azure Storage Explorer is a standalone application that makes it easy to work with Azure Storage data on Windows, macOS, and Linux. With Azure Storage Explorer, you can access multiple accounts and subscriptions and manage all your Storage content.



Things to know about Azure Storage Explorer



Azure Storage Explorer has the following characteristics.

- Azure Storage Explorer requires both management (Azure Resource Manager) and data layer permissions to allow full access to your resources. You need Microsoft Entra ID permissions to access your storage account, the containers in your account, and the data in the containers.
- Azure Storage Explorer lets you connect to different storage accounts.
    - Connect to storage accounts associated with your Azure subscriptions.
    - Connect to storage accounts and services that are shared from other Azure subscriptions.
    - Connect to and manage local storage by using the Azure Storage Emulator.

Things to consider when using Azure Storage Explorer

Azure Storage Explorer supports many scenarios for working with storage accounts in Azure. As you review these options, think about which scenarios apply to your Azure Storage implementation.

Scenario	Description
Connect to an Azure subscription	Manage storage resources that belong to your Azure subscription.
Work with local development storage	Manage local storage by using the Azure Storage Emulator.
Attach to external storage	Manage storage resources that belong to another Azure subscription or that are under national Azure clouds by using the storage account name, key, and endpoints. This scenario is described in more detail in the next section.
Attach a storage account with a SAS	Manage storage resources that belong to another Azure subscription by using a shared access signature (SAS).
Attach a service with a SAS	Manage a specific Azure Storage service (blob container, queue, or table) that belongs to another Azure subscription by using a SAS.

Access keys

Access keys provide access to the entire storage account. You're provided two access keys so you can maintain connections by using one key while regenerating the other.

Important: Store your access keys securely. We recommend regenerating your access keys regularly.

When you regenerate your access keys, you must update any Azure resources and applications that access this storage account to use the new keys. This action doesn't interrupt access to disks from your virtual machines.

Consider Azure File Sync

Azure File Sync enables you to cache several Azure Files shares on an on-premises Windows Server or cloud virtual machine. You can use Azure File Sync to centralize your organization's file shares in Azure Files, while keeping the flexibility, performance, and compatibility of an on-premises file server.

Things to know about Azure File Sync

Let's take a look at the characteristics of Azure File Sync.

- Azure File Sync transforms Windows Server into a quick cache of your Azure Files shares.
- You can use any protocol that's available on Windows Server to access your data locally with Azure File Sync, including SMB, NFS, and FTPS.
- Azure File Sync supports as many caches as you need around the world.

Cloud tiering

Cloud tiering is an optional feature of Azure File Sync. Frequently accessed files are cached locally on the server while all other files are tiered to Azure Files based on policy settings.

- When a file is tiered, Azure File Sync replaces the file locally with a pointer. A pointer is commonly referred to as a *reparse point*. The parse point represents a URL to the file in Azure Files.

- When a user opens a tiered file, Azure File Sync seamlessly recalls the file data from Azure Files without the user needing to know that the file is stored in Azure.
- Cloud tiering files have greyed icons with an offline O file attribute to let the user know when the file is only in Azure.

Things to consider when using Azure File Sync

There are many advantages to using Azure File Sync. Consider the following scenarios and think about how you can use Azure File Sync with your Azure Files shares.

- **Consider application lift and shift**. Use Azure File Sync to move applications that require access between Azure and on-premises systems. Provide write access to the same data across Windows Servers and Azure Files.
- **Consider support for branch offices**. Support your branch offices that need to back up files by using Azure File Sync. Use the service to set up a new server that connects to Azure storage.
- **Consider backup and disaster recovery**. After you implement Azure File Sync, Azure Backup backs up your on-premises data. Restore file metadata immediately and recall data as needed for rapid disaster recovery.
- **Consider file archiving with cloud tiering**. Azure File Sync stores only recently accessed data on local servers. Implement cloud tiering so older data moves to Azure Files.

The main takeaways for this module are:

- Azure Files provides the SMB and NFS protocols, client libraries, and a REST interface that allows access from anywhere to stored files.
- Azure Files is ideal to lift and shift an application to the cloud that already uses the native file system APIs. Share data between the app and other applications running in Azure.
- Azure Files offers two industry-standard file system protocols for mounting Azure file shares: the Server Message Block (SMB) protocol and the Network File System (NFS) protocol.
- Azure Files offers two types of file shares: standard and premium. The premium tier stores data on modern solid-state drives (SSDs), while the standard tier uses hard disk drives (HDDs).
- File share snapshots capture a point-in-time, read-only copy of your data.
- Azure Storage Explorer is a standalone application that makes it easy to work with stored data on Windows, macOS, and Linux.
- Azure File Sync enables you to cache file shares on an on-premises Windows Server or cloud virtual machine.

How many Azure storage accounts you need?

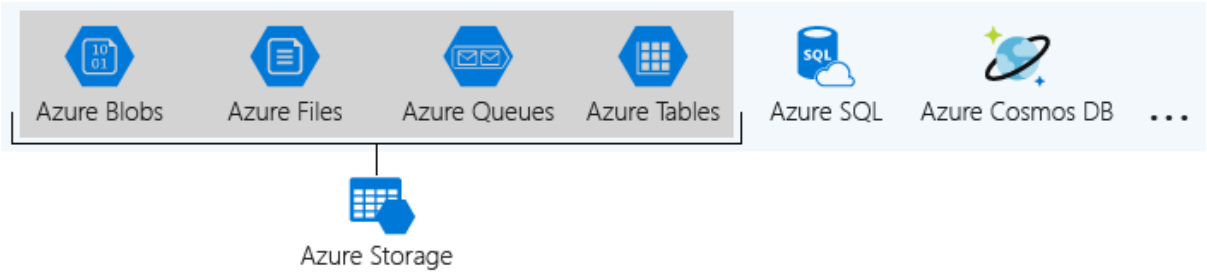
What is Azure Storage?

Azure provides many ways to store your data, including multiple database options like Azure SQL Database, Azure Cosmos DB, and Azure Table Storage. Azure offers multiple ways to store and send messages, such as Azure Queues and Event Hubs. You can even store loose files using services like Azure Files and Azure Blobs.

Azure groups four of these data services together under the name *Azure Storage*. The four services are:

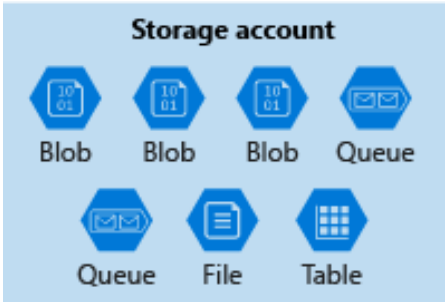
- Azure Blobs
- Azure Files
- Azure Queues
- Azure Tables

The following illustration shows the elements of Azure Storage.



These four data services are all primitive, cloud-based storage services, and are often used together in the same application.

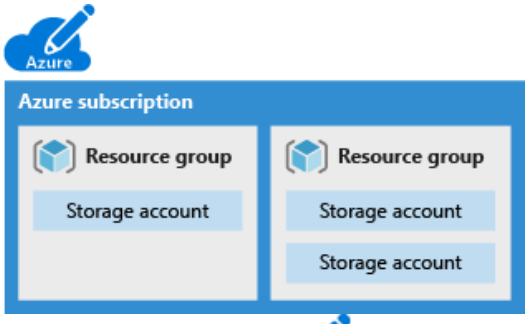
What is a storage account?



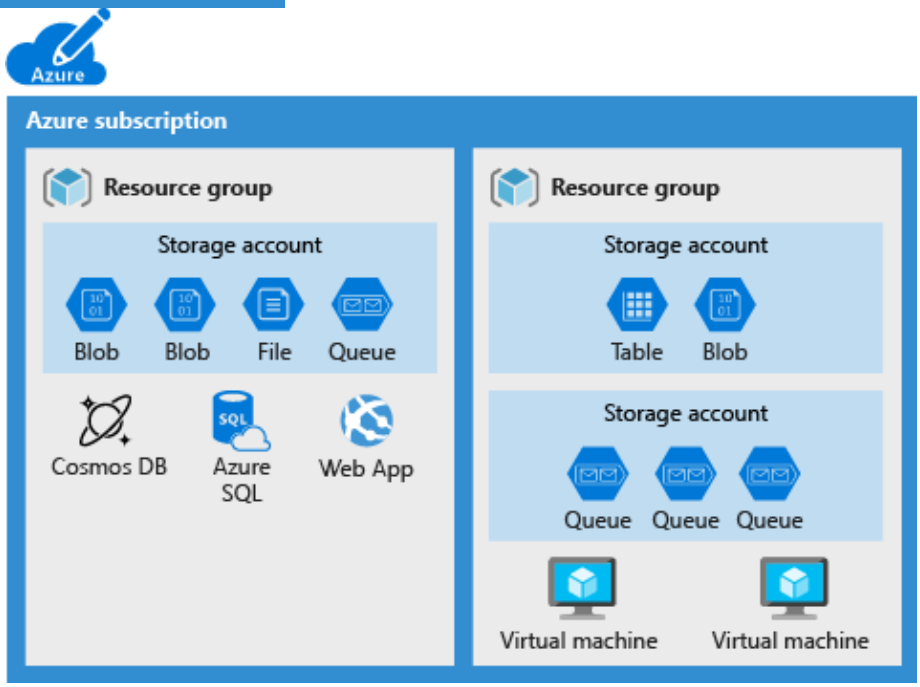
A *storage account* is a container that groups a set of Azure Storage services together. Only data services from Azure Storage can be included in a storage account (Azure Blobs, Azure Files, Azure Queues, and Azure Tables). The following illustration shows a storage account containing several data services.

Combining data services into a single storage account enables you to manage them as a group. The settings you specify when you create the account, or any changes that you make after creation, apply to all services in the storage account. Deleting a storage account deletes all of the data stored inside it.

A storage account is an Azure resource and is part of a resource group. The following illustration shows an Azure subscription containing multiple resource groups, where each group contains one or more storage accounts.



Other Azure data services, such as Azure SQL and Azure Cosmos DB, are managed as independent Azure resources and can't be included in a storage account. The following illustration shows a typical arrangement: Blobs, Files, Queues, and Tables are contained within storage accounts, while other services aren't.



Storage account settings

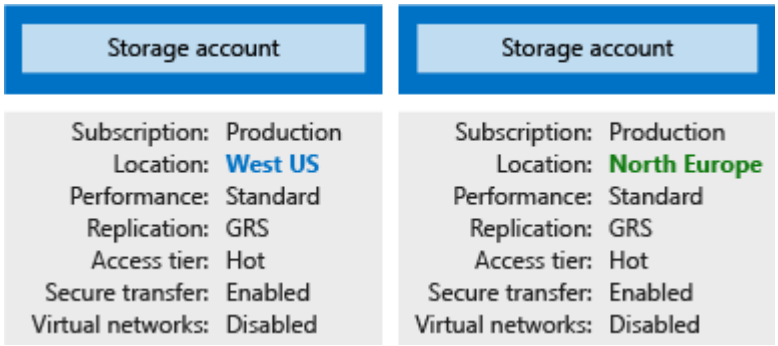
A storage account defines a policy that applies to all the storage services in the account. For example, you could specify that all the contained services will be stored in the West US datacenter, accessible only over https, and billed to the sales department's subscription.

A storage account defines the following settings:

- **Subscription**: The Azure subscription that the storage account services are billed to.
- **Location**: The datacenter that stores the services in the account.
- **Performance**: Determines the data services you can have in your storage account and the type of hardware disks used to store the data.
    - **Standard** allows you to have any data service (Blob, File, Queue, Table) and uses magnetic disk drives.
    - **Premium** provides more services for storing data. For example, storing unstructured object data as block blobs or append blobs, and specialized file storage used to store and create premium file shares. These storage accounts use solid-state drives (SSD) for storage.
- **Replication**: Determines the strategy used to make copies of your data to protect against hardware failure or natural disaster. At a minimum, Azure automatically maintains three copies of your data within the datacenter associated with the storage account. The minimum replication is called locally redundant storage (LRS), and guards against hardware failure but doesn't protect you from an event that incapacitates the entire datacenter. You can upgrade to one of the other options such as geo-redundant storage (GRS) to get replication at different datacenters across the world.
- **Access tier**: Controls how quickly you're able to access the blobs in a storage account. The Hot access tier is optimized for storing frequently accessed or modified data and provides quicker access than Cool, but at increased storage cost. The Cool access tier is optimized for storing infrequently accessed or modified data, and has a lower storage cost. Hot access tier applies only to blobs, and serves as the default value for new blobs.
- **Secure transfer required**: A security feature that determines the supported protocols for access. Enabled requires HTTPS, while disabled allows HTTP.
- **Virtual networks**: A security feature that allows inbound access requests from only the virtual networks that you specify.

How many storage accounts do you need?

A storage account represents a collection of settings like location, replication strategy, and subscription owner. You need one storage account for each group of settings that you want to apply to your data. The following illustration shows two storage accounts that differ in one setting; that one difference is enough to require separate storage accounts.



Typically, your data diversity, cost sensitivity, and tolerance for management overhead determine the number of storage accounts you need.

Data diversity

Organizations often generate data that differs along several vectors. For example, where the data is consumed, how sensitive it is, which group pays the bills for it, etc. Diversity along any of these vectors can lead to multiple storage accounts. Let's consider two examples:

1. Do you have data that is specific to a country/region? If so, you might want to store the data in a datacenter in that country/region for performance or compliance reasons. You need one storage account for each geographical region.
2. Do you have some data that is proprietary and some for public consumption? If so, you could enable virtual networks for the proprietary data and not for the public data. Separating proprietary data and public data requires separate storage accounts.

In general, increased diversity means an increased number of storage accounts.

Cost sensitivity

A storage account by itself has no financial cost; however, the settings you choose for the account do influence the cost of services in the account. Geo-redundant storage costs more than locally redundant storage. Premium performance and the Hot access tier increase the cost of blobs.

You can use multiple storage accounts to reduce costs. For example, you could partition your data into critical and noncritical categories. You could place your critical data into a storage account with geo-redundant storage and put your noncritical data in a different storage account with locally redundant storage.

Tolerance for management overhead

Each storage account requires some time and attention from an administrator to create and maintain. It also increases complexity for anyone who adds data to your cloud storage. Everyone in an administrator role needs to understand the purpose of each storage account so they add new data to the correct account.

Storage accounts are powerful tools to help you obtain the performance and security you need while minimizing costs. A typical strategy is to start with an analysis of your data. Create partitions that share characteristics like location, billing, and replication strategy. Then, create one storage account for each partition.

Account kind

Storage account *kind* is a set of policies that determine which data services you can include in the account and the pricing of those services. There are four kinds of storage accounts:

- **Standard - StorageV2 (general purpose v2)**: the current offering that supports all storage types and all of the latest features.
- **Premium - Page blobs**: Premium storage account type for page blobs only.
- **Premium - Block blobs**: Premium storage account type for block blobs and append blobs.
- **Premium - File shares**: Premium storage account type for file shares only.

Microsoft recommends that you use the **Standard - StorageV2 (general purpose v2)** option for new storage accounts.

The core advice is to choose the **Resource Manager** deployment model and the **Standard - StorageV2 (general purpose v2)** account kind for all your storage accounts. For new resources, there are few reasons to consider the other choices.

Plan for maintenance and downtime

Azure Administrators prepare for planned and unplanned failures.

Things to know about maintenance planning

An availability plan for Azure virtual machines needs to include strategies for unplanned hardware maintenance, unexpected downtime, and planned maintenance.

- An **unplanned hardware maintenance** event occurs when the Azure platform predicts that the hardware or any platform component associated to a physical machine is about to fail. When the platform predicts a failure, it issues an unplanned hardware maintenance event. Azure uses Live Migration technology to migrate your virtual machines from the failing hardware to a healthy physical machine. Live Migration is a virtual machine preserving operation that only pauses the virtual machine for a short time, but performance might be reduced before or after the event.
- **Unexpected downtime** occurs when the hardware or the physical infrastructure for your virtual machine fails unexpectedly. Unexpected downtime can include local network failures, local disk failures, or other rack level failures. When detected, the Azure platform automatically migrates (heals) your virtual machine to a healthy physical machine in the same datacenter. During the healing procedure, virtual machines experience downtime (reboot) and in some cases loss of the temporary drive.
- **Planned maintenance** events are periodic updates made by Microsoft to the underlying Azure platform to improve overall reliability, performance, and security of the platform infrastructure that your virtual machines run on.

Create availability sets

An availability set is a logical feature you can use to ensure a group of related virtual machines are deployed together. The grouping helps to prevent a single point of failure from affecting all of your machines. The grouping ensures that not all of the machines are upgraded at the same time during a host operating system upgrade in the datacenter.

Things to know about availability sets

Let's review some characteristics of availability sets.

- All virtual machines in an availability set should perform the identical set of functionalities.
- All virtual machines in an availability set should have the same software installed.
- Azure ensures that virtual machines in an availability set run across multiple physical servers, compute racks, storage units, and network switches.

If a hardware or Azure software failure occurs, only a subset of the virtual machines in the availability set are affected. Your application stays up and continues to be available to your customers.

- You can create a virtual machine and an availability set at the same time.

A virtual machine can only be added to an availability set when the virtual machine is created. To change the availability set for a virtual machine, you need to delete and then recreate the virtual machine.

- You can build availability sets by using the Azure portal, Azure Resource Manager (ARM) templates, scripting, or API tools.
- Microsoft provides robust Service Level Agreements (SLAs) for Azure virtual machines and availability sets. For details, see SLA for Azure Virtual Machines.

Things to consider when using availability sets

Availability sets are an essential capability when you want to build reliable cloud solutions. In your planning for availability sets, keep the following general principles in mind:

- **Consider redundancy**. To achieve redundancy in your configuration, place multiple virtual machines in an availability set.
- **Consider separation of application tiers**. Each application tier exercised in your configuration should be located in a separate availability set. The separation helps to mitigate single point of failure on all machines.
- **Consider load balancing**. For high availability and network performance, create a load-balanced availability set by using Azure Load Balancer. Load Balancer distributes incoming traffic across working instances of services that are defined in your load-balanced availability set.
- **Consider managed disks**. You can use Azure managed disks with your Azure virtual machines in availability sets for block-level storage.

Update Domains and Fault Domains

Azure Virtual Machine Availability Sets implements two node concepts to help Azure maintain high availability and fault tolerance when deploying and upgrading applications: *update domains* and *fault domains*. Each virtual machine in an availability set is placed in one update domain and one fault domain.

Things to know about update domains

An update domain is a group of nodes that are upgraded together during the process of a service upgrade (or *roll out*). An update domain allows Azure to perform incremental or rolling upgrades across a deployment. Here are some other characteristics of update domains.

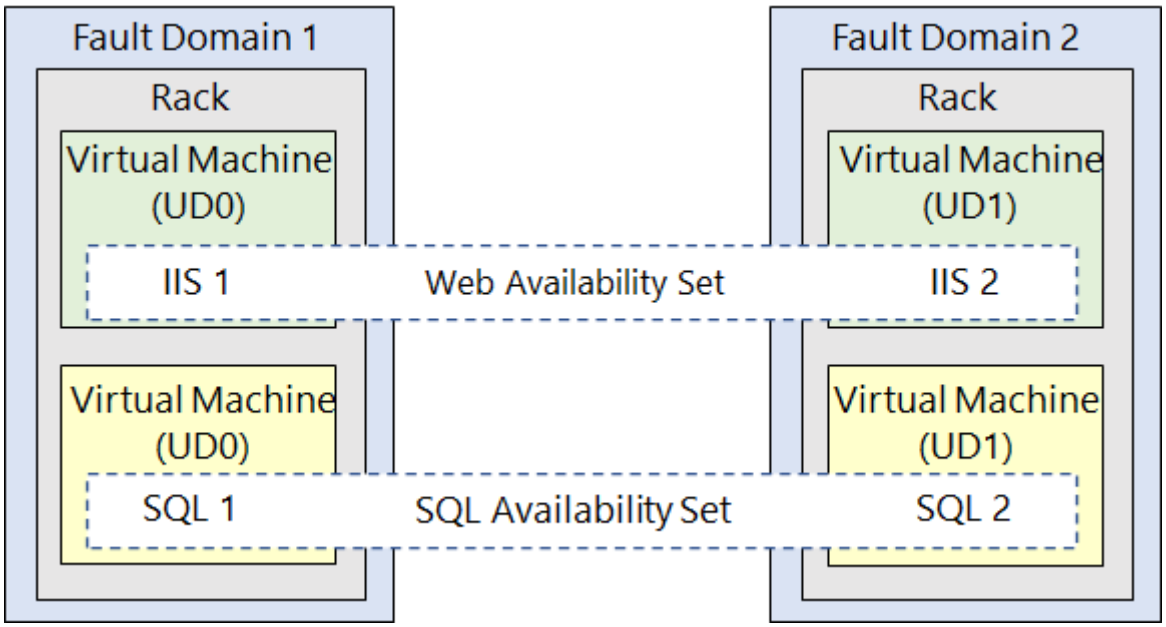
- Each update domain contains a set of virtual machines and associated physical hardware that can be updated and rebooted at the same time.
- During planned maintenance, only one update domain is rebooted at a time.
- By default, there are five (non-user-configurable) update domains.
- You can configure up to 20 update domains.

Things to know about fault domains

A fault domain is a group of nodes that represent a physical unit of failure. Think of a fault domain as nodes that belong to the same physical rack.

- A fault domain defines a group of virtual machines that share a common set of hardware (or *switches*) that share a single point of failure. An example is a server rack serviced by a set of power or networking switches.
- Two fault domains work together to mitigate against hardware failures, network outages, power interruptions, or software updates.

Let's look at a scenario with two fault domains that have two virtual machines each. The virtual machines in each fault domain are contained in different availability sets. The web availability set contains two virtual machines with one machine from each fault domain. The SQL availability set contains two different virtual machines with one from each fault domain.



Availability Zones (AZ)

Availability zones are a high-availability offering that protects your applications and data from datacenter failures. An availability zone in an Azure region is a combination of a fault domain and an update domain.

Consider a scenario where you create three or more virtual machines across three zones in an Azure region. Your virtual machines are effectively distributed across three fault domains and three update domains. The Azure platform recognizes this distribution across update domains to make sure that virtual machines in different zones aren't updated at the same time.

You can use availability zones to build high-availability into your application architecture by colocating your compute, storage, networking, and data resources within a zone and replicating in other zones.

Things to know about availability zones

Review the following characteristics of availability zones.

- Availability zones are unique physical locations within an Azure region.
- Each zone is made up of one or more datacenters that are equipped with independent power, cooling, and networking.
- To ensure resiliency, there's a minimum of three separate zones in all enabled regions.
- The physical separation of availability zones within a region protects applications and data from datacenter failures.
- Zone-redundant services replicate your applications and data across availability zones to protect against single-points-of-failure.

Things to consider when using availability zones

Azure services that support availability zones are divided into two categories.

Category	Description	Examples
Zonal services	Azure *zonal* services pin each resource to a specific zone.	- Azure Virtual Machines - Azure managed disks - Standard IP addresses
Zone-redundant services	For Azure services that are zone-redundant, the platform replicates automatically across all zones.	- Azure Storage that's zone-redundant - Azure SQL Database

Compare vertical and horizontal scaling

A robust virtual machine configuration includes support for scalability. Scalability allows throughput for a virtual machine in proportion to the availability of the associated hardware resources. A scalable virtual machine can handle increases in requests without adversely affecting response time and throughput. For most scaling operations, there are two implementation options: *vertical* and *horizontal*.

Things to know about vertical scaling

Vertical scaling, also known as *scale up and scale down*, involves increasing or decreasing the virtual machine **size** in response to a workload. Vertical scaling makes a virtual machine more (scale up) or less (scale down) powerful.

Here are some scenarios where using vertical scaling can be advantageous:

- If you have a service built on a virtual machine that's under-utilized such as on the weekend, you can use vertical scaling to decrease the virtual machine size and reduce your monthly costs.
- You can implement vertical scaling to increase your virtual machine size to support larger demand without having to create extra virtual machines.

Things to know about horizontal scaling

Horizontal scaling, also referred to as *scale out and scale in*, is used to adjust the **number** of virtual machines in your configuration to support the changing workload. When you implement horizontal scaling, there's an increase (scale out) or decrease (scale in) in the number of virtual machine instances.

Things to consider when using vertical and horizontal scaling

Review the following considerations regarding vertical and horizontal scaling. Think about which implementation might be required to support your company website.

- **Consider limitations**. Generally speaking, horizontal scaling has fewer limitations than vertical scaling. A vertical scaling implementation depends on the availability of larger hardware, which quickly hits an upper

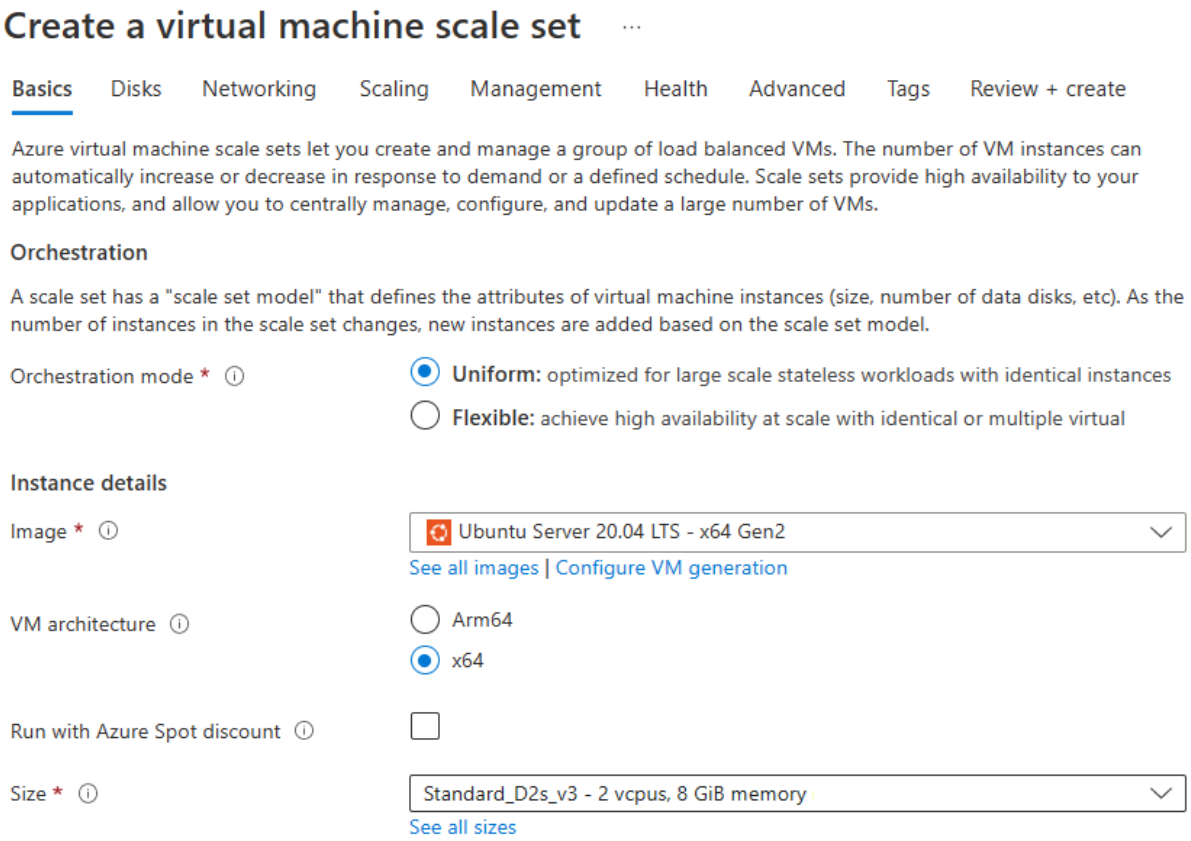
limit and can vary by region. Vertical scaling also usually requires a virtual machine to stop and restart, which can temporarily limit access to applications or data.

- **Consider flexibility**. When operating in the cloud, horizontal scaling is more flexible. A horizontal scaling implementation allows you to run potentially thousands of virtual machines to manage changes in workload and throughput.
- **Consider reprovisioning**. *Reprovisioning* is the process of removing an existing virtual machine and replacing it with a new machine. A robust availability plan considers where reprovisioning might be required and plans for interruptions to service. If reprovisioning might be required, determine if any data needs to be maintained and migrated to the new machine.

Virtual Machine Scale Sets

You can implement Azure Virtual Machine Scale Sets in the Azure portal. You specify the number of virtual machines and their sizes, and indicate preferences for using Azure Spot instances, Azure managed disks, and allocation policies.

In the Azure portal, there are several settings to configure to create an Azure Virtual Machine Scale Sets implementation.

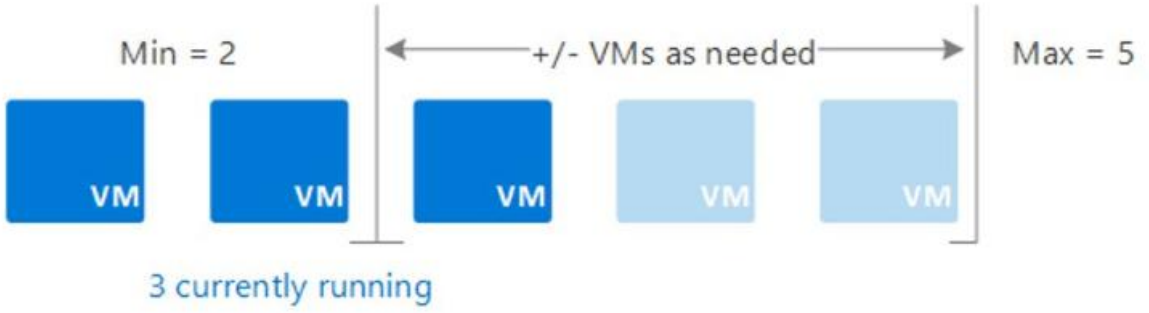


- [Orchestration mode]: Choose how the scale set manages the virtual machines. In flexible orchestration mode, you manually create and add a virtual machine of any configuration to the scale set. In uniform orchestration mode, you define a virtual machine model and Azure generates identical instances based on that model.
- **Image**: Choose the base operating system or application for the VM.
- **VM Architecture**: Azure provides a choice of x64 or Arm64-based virtual machines to run your applications. x64-based VMs provide the most software compatibility. Arm64-based VMs provide up to 50% better price-performance than comparable x64 VMs.

- **Size**: Select a VM size to support the workload that you want to run. The size that you choose then determines factors such as processing power, memory, and storage capacity. Azure offers a wide variety of sizes to support many types of uses. Azure charges an hourly price based on the VM's size and operating system.
- **Spreading algorithm**: The spreading algorithm determines how VMs in the scale set are balanced across fault domains. With max spreading, VMs are spread across as many fault domains as possible in each zone. With fixed spreading, VMs are always spread across exactly five fault domains. In the case where fewer than five fault domains are available, a scale set using "Max spreading" completes, while a scale set using "Fixed spreading" fails. For this reason, Microsoft recommends using **Max spreading** for your implementation.

Implement autoscale

An Azure Virtual Machine Scale Sets implementation can automatically increase or decrease the number of virtual machine instances that run your application. This process is known as *autoscaling*. Autoscaling allows you to dynamically scale your configuration to meet changing workload demands.



Autoscaling minimizes the number of unnecessary virtual machine instances that run your application when demand is low. Your customers continue to receive an acceptable level of performance as demand grows and more virtual machine instances are automatically added.

Things to consider when using autoscaling

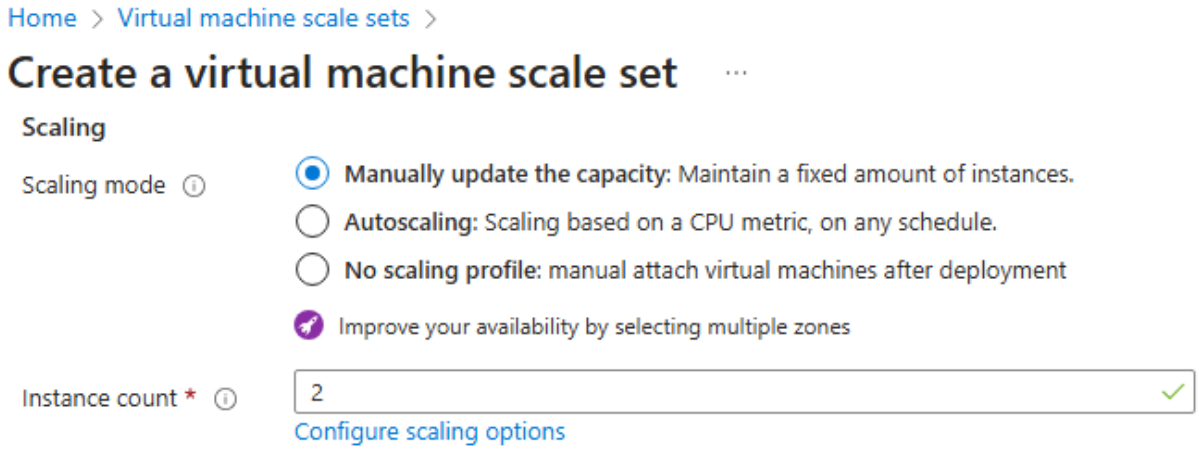
Review the following considerations about autoscaling. Think about how this process can benefit your company website implementation.

- **Consider automatic adjusted capacity**. You can create autoscaling rules to define the acceptable performance for a positive customer experience. When the defined thresholds are met, the autoscale rules act to adjust the capacity of your Virtual Machine Scale Sets implementation.
- **Consider scale out**. If your application demand increases, the load on the virtual machine instances in your implementation increases. If the increased load is consistent, rather than a brief demand, you can configure autoscale rules to increase the number of virtual machine instances in your implementation.
- **Consider scale in**. On an evening or weekend, your application demand might decrease. If the decreased load is consistent over a period of time, you can configure autoscale rules to decrease the number of virtual machine instances in your implementation. The scale-in action reduces the cost to run your Virtual Machine Scale Sets implementation as you only run the number of instances required to meet the current demand.
- **Consider scheduled events**. You can implement autoscaling and schedule events to automatically increase or decrease the capacity of your implementation at fixed times.
- **Consider overhead**. Using Azure Virtual Machine Scale Sets with autoscaling reduces your management overhead to monitor and optimize the performance of your application.

Configure autoscale

When you create an Azure Virtual Machine Scale Sets implementation in the Azure portal, you can enable manual or autoscaling. For optimal performance, you should define a minimum, maximum, and default number of virtual machine instances to use.

In the Azure portal, you can select the scaling mode.

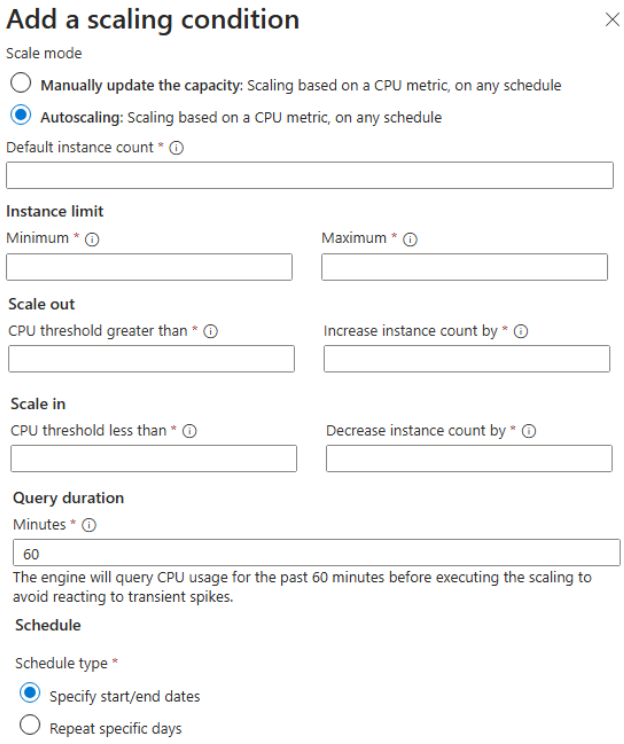


Scaling mode

- **Manually update the capacity**: You can manually update the capacity and maintain a fixed instances count. Set the **Instance count** to the number of virtual machines in the scale set (0 - 1000). Configure the **Scale-in policy** which is the order virtual machines are selected for deletion. For example, you could balance across zones and then delete the virtual machine with the highest instance ID.
- **Autoscaling**: You can autoscale the capacity on any schedule, based on any metrics. Specify the maximum number of virtual machines that can be available when autoscaling is applied on your implementation.

Configure autoscaling

Autoscaling is based on a scaling condition.



- **Default instance count.** The initial number of virtual machines deployed in this scale set (0-1000).
- **Instance limit.** The minimum instance count you want this condition to scale down to. The maximum instance count you want this condition to scale up to.
- **Scale out.** The CPU usage percentage threshold for triggering the scale-out autoscale rule. The number of instances to scale out by.
- **Scale in.** The CPU usage percentage threshold for triggering the scale-in autoscale rule. The number of instances to scale in by.
- **Query duration**: This duration is the time the Autoscale engine looks back for the metric usage average. This look back is to allow your metric to stabilize.
- **Schedule**: Specify the start and end dates. You can also repeat the schedule on specific days.

Implement Azure App Service plans

In Azure App Service, an application runs in an Azure App Service plan. An App Service plan defines a set of compute resources for a web application to run. The compute resources are analogous to a server farm in conventional web hosting. One or more applications can be configured to run on the same computing resources (or in the same App Service plan).

Things to know about App Service plans

Let's take a closer look at how to implement and use an App Service plan with your virtual machines.

- When you create an App Service plan in a region, a set of compute resources is created for the plan in the specified region. Any applications that you place into the plan run on the compute resources defined by the plan.
- Each App Service plan defines these settings:
    - **Operating system**: Linux or Windows.
    - **Region**: The region for the App Service plan, such as West US, Central India, North Europe, and so on.
    - **Pricing tier**: Determines what App Service features you get and how much you pay for the plan. The pricing tiers available to your App Service plan depend on the operating system selected at creation time.
    - **Number of VM instances**: Currently ranges from three to 30.
    - **Size of VM instances**: Defined by CPU, memory, and remote storage.
- You can continue to add new applications to an existing plan as long as the plan has enough resources to handle the increasing load.

Things to consider when using App Service plans

Review the following considerations about using Azure App Service plans to run and scale your applications. Think about what conditions might apply to running and scaling the hotel website.

- **Consider cost savings**. Because you pay for the computing resources that your App Service plan allocates, you can potentially save money by placing multiple applications into the same App Service plan.
- **Consider multiple applications in one plan**. Create a single plan to support multiple applications, to make it easier to configure and maintain shared virtual machine instances. Because the applications share the same virtual machine instances, you need to carefully manage your plan resources and capacity.
- **Consider plan capacity**. Before you add a new application to an existing plan, determine the resource requirements for the new application and identify the remaining capacity of your plan.

Important

Overloading an App Service plan can potentially cause downtime for new and existing applications.

- **Consider application isolation**. Isolate your application into a new App Service plan when:
    - The application is resource intensive.
    - You want to scale the application independently from the other applications in the existing plan.
    - The application needs resource in a different geographical region.

Azure App Service plan pricing

The pricing tier of an Azure App Service plan determines what App Service features you get and how much you pay for the plan. Pricing tier examples are: Free, Shared, Basic, Standard, Premium, PremiumV2, PremiumV3, Isolated, and IsolatedV2.

How applications run and scale in App Service plans

The Azure App Service plan is the scale unit of App Service applications. Depending on the pricing tier for your Azure App Service plan, your applications run and scale in a different manner. If your plan is configured to run five virtual

machine instances, then all applications in the plan run on all five instances. If your plan is configured for autoscaling, then all applications in the plan are scaled out together based on the autoscale settings.

The pricing tiers are grouped into three categories:

- **Shared compute**:
    - Free and Shared, the two base tiers, run an app on the same Azure VM as other App Service apps, including apps of other customers.
    - These tiers allocate CPU quotas to each app that runs on the shared resources, and the resources can't scale out.
    - These tiers are intended to be used only for development and testing purposes.
- **Dedicated compute**:
    - The Basic, Standard, Premium, PremiumV2, and PremiumV3 tiers run apps on dedicated Azure VMs.
    - Only apps in the same App Service plan have the same compute resources. The higher the tier, the more VM instances that are available to you for scale-out.
- **Isolated**:
    - The Isolated and IsolatedV2 tiers run dedicated Azure VMs on dedicated Azure virtual networks.
    - This tier provides network isolation on top of compute isolation to your apps.
    - This tier provides the maximum scale-out capabilities.

Here's a sample of different plan details. For more information, you can use the Azure portal. Select *Create App Service Plan* and then *Explore pricing plans*. You can view either by hardware or feature capability.

Feature	Free F1	Basic B1	Standard S1	Premium P1V3
Usage	Development, Testing	Development, Testing	Production workloads	Enhanced scale, performance
Staging slots	N/A	N/A	5	20
Auto scale	N/A	Manual	Rules	Rules, Elastic
Scale instances	N/A	3	10	30
Daily backups	N/A	N/A	10	50

Free and Shared

The Free and Shared service plans are base tiers that run on the same Azure virtual machines as other applications. Some applications might belong to other customers. These tiers are intended to be used for development and testing purposes only. No SLA is provided for the Free and Shared service plans. Free and Shared plans are metered on a per application basis.

Basic

The Basic service plan is designed for applications that have lower traffic requirements, and don't need advanced auto scale and traffic management features. Pricing is based on the size and number of instances you run. Built-in network load-balancing support automatically distributes traffic across instances. The Basic service plan with Linux runtime environments supports Web App for Containers.

Standard

The Standard service plan is designed for running production workloads. Pricing is based on the size and number of instances you run. Built-in network load-balancing support automatically distributes traffic across instances. The Standard plan includes auto scale that can automatically adjust the number of virtual machine instances running to match your traffic needs. The Standard service plan with Linux runtime environments supports Web App for Containers.

Premium

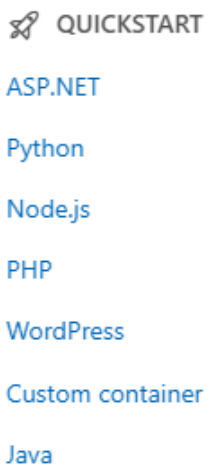
The Premium service plan is designed to provide enhanced performance for production applications. The upgraded Premium plan, Premium v2, offers Dv2-series virtual machines with faster processors, SSD storage, and double memory-to-core ratio compared to the Standard tier. The new Premium plan also supports higher scale via increased instance count while still providing all the advanced capabilities of the Standard tier. The first generation of Premium plan is still available to support existing customer scaling needs.

Isolated

The Isolated service plan is designed to run mission critical workloads that are required to run in a virtual network. The Isolated plan allows customers to run their applications in a private, dedicated environment in an Azure datacenter. The plan offers Dv2-series virtual machines with faster processors, SSD storage, and a double memory-to-core ratio compared to the Standard tier. The private environment used with an Isolated plan is called the App Service Environment. The plan can scale to 100 instances with more available upon request.

Implement Azure App Service

Azure App Service brings together everything you need to create websites, mobile backends, and web APIs for any platform or device. Applications run and scale with ease in both Windows and Linux-based environments.



Things to know about configuration settings

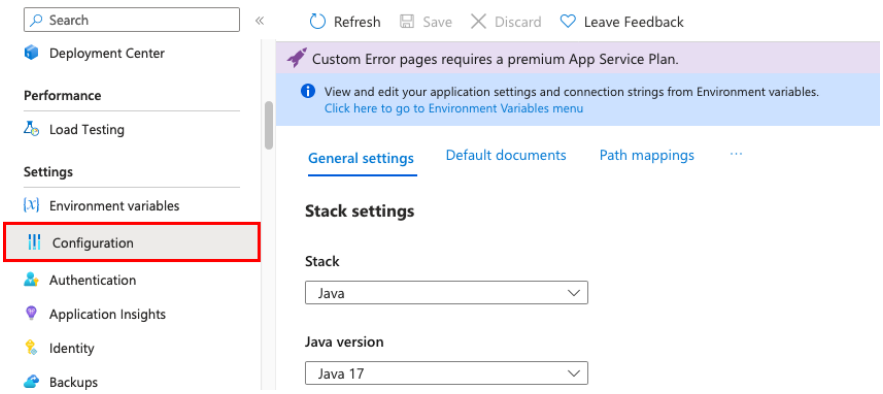
Let's examine some of the basic configuration settings you need to create an app with App Service.

- **Name**: The name for your app must be unique. The name identifies and locates your app in Azure. An example name is webappces1.azurewebsites.net. You can map a custom domain name, if you prefer to use that option instead.
- **Publish**: App Service hosts (publishes) your app as code or as a Docker Container.
- **Runtime stack**: App Service uses a software stack to run your app, including the language and SDK versions. For Linux apps and custom container apps, you can set an optional start-up command or file. Your choices for the stack include .NET Core, .NET Framework, Node.js, PHP, Python, and Ruby. Various versions of each product are available for Linux and Windows.
- **Operating system**: The operating system for your app runtime stack can be Linux or Windows.
- **Region**: The region location that you choose for your app affects the App Service plans that are available.
- **Pricing plans**: Your app needs to be associated with an Azure App Service plan to establish available resources, features, and capacity. You can choose from pricing tiers that are available for the region location you selected.

Post-creation settings

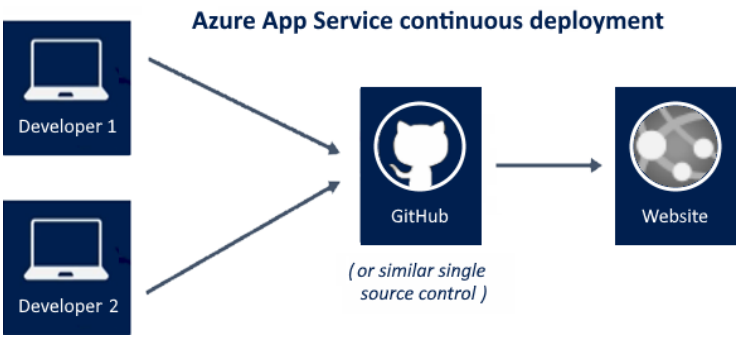
After your app is created, other Configuration settings become available in the Azure portal, including app deployment options and path mapping.

Some of the extra configuration settings can be included in the developer's code, while others can be configured in your app. Here are a few of the extra application settings.



- **Always On**: You can keep your app loaded even when there's no traffic. This setting is required for continuous WebJobs or for WebJobs that are triggered by using a CRON expression.
- **Session affinity**: In a multi-instance deployment, you can ensure your app client is routed to the same instance for the life of the session.
- **HTTPS Only**: When enabled, all HTTP traffic is redirected to HTTPS.

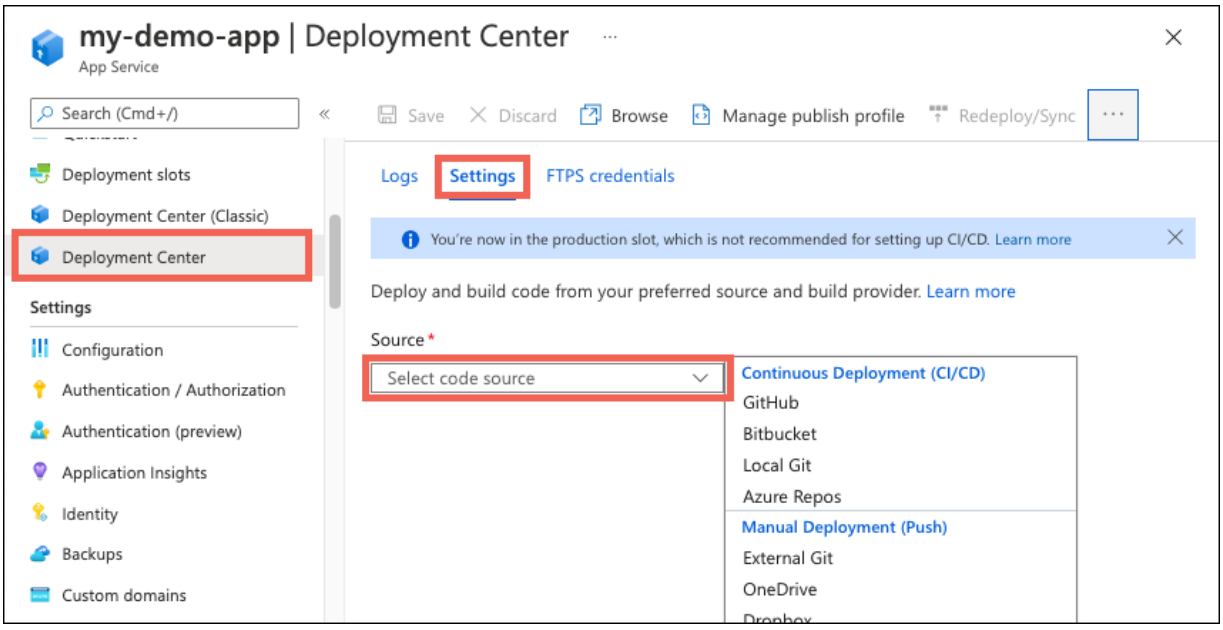
Explore continuous integration and deployment



The Azure portal provides out-of-the-box continuous integration and deployment with Azure DevOps services, GitHub, Bitbucket, FTP, or a local Git repository on your development machine. You can connect your web app with any of the above sources and App Service handles the rest for you. App Service auto synchronizes your code and any future changes to the code into your web app. With Azure DevOps services, you can also define your own build and release process. Compile your source code, run tests, and build and deploy the release into your web app every time you commit the code. All of the operations happen implicitly without any need for human administration.

Things to know about continuous and manual deployment

When you create your web app with App Service, you can choose continuous or manual deployment. As you review these options, consider which deployment method to implement for your App Service apps. These options are located in the Deployment Centre.



Continuous deployment (CI/CD) is a process used to push out new features and bug fixes in a fast and repetitive pattern with minimal impact on end users. Azure supports automated deployment directly from several sources:

- **GitHub**: Azure supports automated deployment directly from GitHub. When you connect your GitHub repository to Azure for automated deployment, any changes you push to your production branch on GitHub are automatically deployed for you.
- **Bitbucket**: With its similarities to GitHub, you can configure an automated deployment with Bitbucket.
- **Local Git**: The App Service Web Apps feature offers a local URL that you can add as a repository.
- **Azure Repos**: Azure Repos is a set of version control tools that you can use to manage your code. Whether your software project is large or small, using version control as soon as possible is a good idea.

Manual deployment enables you to manually push your code to Azure. There are several options for manually pushing your code:

- **Remote Git**: The App Service Web Apps feature offers a Git URL that you can add as a remote repository. Pushing to the remote repository deploys your app.
- **OneDrive**: OneDrive is a service that lets you store and share files on the internet with a Microsoft account.
- **Dropbox**: Dropbox is a file hosting service

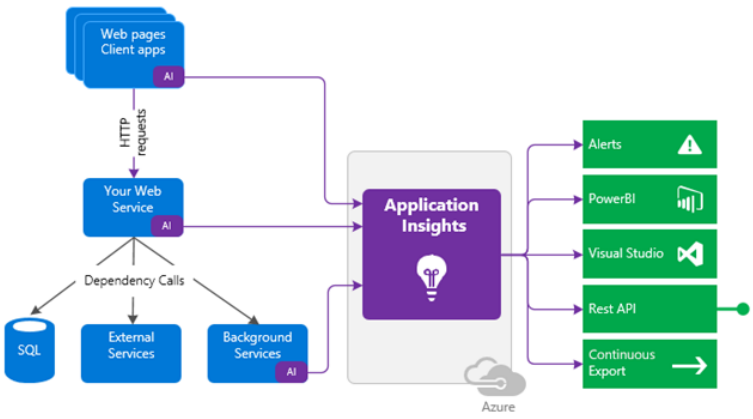
Use Azure Application Insights

Azure Application Insights is a feature of Azure Monitor that lets you monitor your live applications. You can integrate Application Insights with your App Service configure to automatically detect performance anomalies in your apps.

Application Insights is designed to help you continuously improve the performance and usability of your apps. The feature offers powerful analytics tools to help you diagnose issues and understand what users actually do with your apps.

Things to know about Application Insights

Let's examine some characteristics of Application Insights for Azure Monitor.



- Application Insights works on various platforms including .NET, Node.js and Java EE.
- The feature can be used for configurations that are hosted on-premises, in a hybrid environment, or in any public cloud.
- Application Insights integrates with your Azure Pipeline processes and has connection points to many development tools.
- You can monitor and analyse data from mobile apps by integrating with Visual Studio App Center.

Things to consider when using Application Insights

Application Insights is ideal for supporting your development team. The feature helps developers understand how your app is performing and how it's being used. Consider monitoring the following items in your App Service configuration scenario.

- **Consider Request rates, response times, and failure rates**. Find out which pages are most popular, at what times of day, and where your users are. See which pages perform best. If your response times and failure rates go high when there are more requests, then perhaps you have a resourcing problem.
- **Consider Dependency rates, response times, and failure rates**. Use Application Insights to discover if external services are degrading your app performance.
- **Consider Exceptions**. Analyze the aggregated statistics or pick specific instances and drill into the stack trace and related requests. Both server and browser exceptions are reported.
- **Consider Page views and load performance**. Collect the number of page views reported by your users' browsers and analyze the load performance.
- **Consider User and session counts**. Application Insights can help you keep track of the number of users and sessions connected to your app.
- **Consider Performance counters**. Add Application Insights performance counters from your Windows or Linux server machines. Monitor performance output for the CPU, memory, network usage, and so on.
- **Consider Host diagnostics**. Integrate diagnostics from Docker or Azure into your app Application Insights.
- **Consider Diagnostic trace logs**. Implement trace logs from your app to help correlate trace events with requests and diagnose issues.
- **Consider Custom events and metrics**. Write your own custom events and metric tracking algorithms as client or server code. Track business events such as number of items sold, or number of games won.

Azure Container Instances

Hardware virtualization makes it possible to run multiple isolated instances of operating systems concurrently on the same physical hardware. Containers represent the next stage in the virtualization of computing resources.

Container-based virtualization allows you to virtualize the operating system. This approach lets you run multiple applications within the same instance of an operating system, while maintaining isolation between the applications. The containers within a virtual machine provide functionality similar to that of virtual machines within a physical server.

Things to know about containers versus virtual machines

To better understand container-based virtualization, let's compare containers and virtual machines.

Compare	Containers	Virtual machines
Isolation	A container typically provides lightweight isolation from the host and other containers, but a container doesn't provide as strong a security boundary as a virtual machine.	A virtual machine provides complete isolation from the host operating system and other virtual machines. This separation is useful when a strong security boundary is critical, such as hosting apps from competing companies on the same server or cluster.
Operating system	Containers run the user mode portion of an operating system and can be tailored to contain just the needed services for your app. This approach helps you use fewer system resources.	Virtual machines run a complete operating system including the kernel, which requires more system resources (CPU, memory, and storage).
Deployment	You can deploy individual containers by using Docker via the command line. You can deploy multiple containers by using an orchestrator such as Azure Kubernetes Service.	You can deploy individual virtual machines by using Windows Admin Center or Hyper-V Manager. You can deploy multiple virtual machines by using PowerShell or System Center Virtual Machine Manager.
Persistent storage	Containers use Azure Disks for local storage for a single node, or Azure Files (SMB shares) for storage shared by multiple nodes or servers.	Virtual machines use a virtual hard disk (VHD) for local storage for a single machine, or an SMB file share for storage shared by multiple servers.
Fault tolerance	If a cluster node fails, the orchestrator on another cluster node rapidly recreates any containers running on the node.	Virtual machines can fail over to another server in a cluster, where the virtual machine's operating system restarts on the new server.

Things to consider when using containers

Containers offer several advantages over physical and virtual machines. Review the following benefits and consider how you can implement containers for the internal apps for your company.

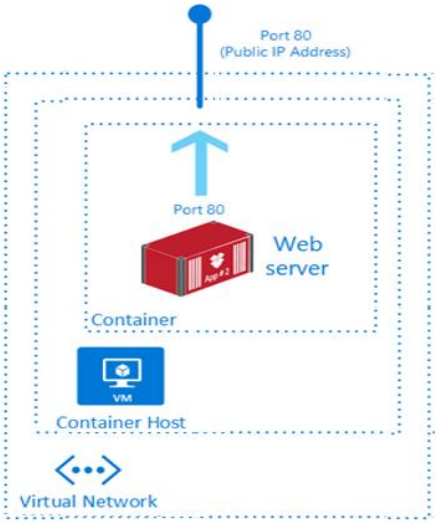
- **Consider flexibility and speed**. Gain increased flexibility and speed when developing and sharing your containerized application code.
- **Consider testing**. Choose containers for your configuration to allow for simplified testing of your apps.
- **Consider app deployment**. Implement containers to gain streamlined and accelerated deployment of your apps.
- **Consider workload density**. Support higher workload density and improve your resource utilization by working with containers.

Azure Container Instances

Containers are becoming the preferred way to package, deploy, and manage cloud applications. There are many options for teams to build and deploy cloud native and containerized applications on Azure. In this unit, we review Azure Container Instances (ACI).

Azure Container Instances offers the fastest and simplest way to run a container in Azure, without having to manage any virtual machines and without having to adopt a higher-level service. Azure Container Instances is a great solution for any scenario that can operate in isolated containers.

The following illustration shows a web server container built with Azure Container Instances. The container is running on a virtual machine in a virtual network.



Understand container images

All containers are created from container images. A container image is a lightweight, standalone, executable package of software that encapsulates everything needed to run an application. It includes the following components:

- **Code**: The application's source code.
- **Runtime**: The environment required to execute the application.
- **System tools**: Utilities necessary for the application to function.
- **System libraries**: Shared libraries used by the application.
- **Settings**: Configuration parameters specific to the application.

When you create a container image, it becomes a portable unit that can run consistently across different computing environments. These images are the building blocks for containers, which are instances of these images running at runtime.

Things to know about Azure Container Instances

Let's review some of the benefits of using Azure Container Instances. As you review these points, think about how you can implement Container Instances for your internal applications.

- **Fast startup times**. Containers can start in seconds without the need to provision and manage virtual machines.
- **Public IP connectivity and DNS names**. Containers can be directly exposed to the internet with an IP address and FQDN (fully qualified domain name).
- **Custom sizes**. Container nodes can be scaled dynamically to match actual resource demands for an application.
- **Persistent storage**. Containers support direct mounting of Azure Files file shares.
- **Linux and Windows containers**. Container Instances can schedule both Windows and Linux containers. Specify the operating system type when you create your container groups.
- **Co-scheduled groups**. Container Instances supports scheduling of multi-container groups that share host machine resources.
- **Virtual network deployment**. Container Instances can be deployed into an Azure virtual network.

Implement container groups

The top-level resource in Azure Container Instances is the **container group**. A container group is a collection of containers that get scheduled on the same host machine. The containers share a lifecycle, resources, local network, and storage volumes.

Things to know about container groups

Let's review some of details about container groups for Azure Container Instances.

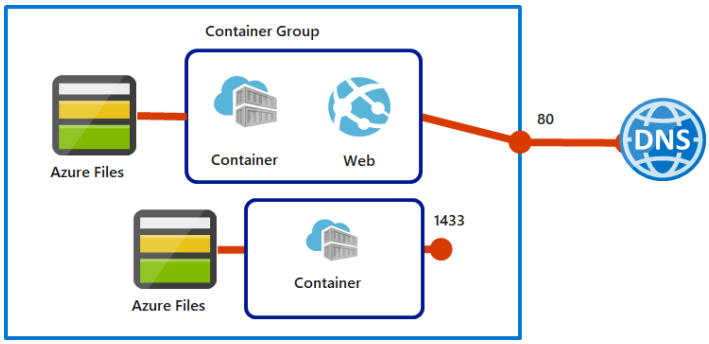
- A container group is similar to a pod in Kubernetes. A pod typically has a 1:1 mapping with a container, but a pod can contain multiple containers. The containers in a multi-container pod can share related resources.
- Azure Container Instances allocates resources to a multi-container group by adding together the resource requests of all containers in the group. Resources can include items such as CPUs, memory, and GPUs.

Consider a container group that has two containers that each require CPU resources. Each container requests one CPU. Azure Container Instances allocates two CPUs for the container group.

- There are two common ways to deploy a multi-container group: Azure Resource Manager (ARM) templates and YAML files.
    - **ARM template**. An ARM template is recommended for deploying other Azure service resources when you deploy your container instances, such as an Azure Files file share.
    - **YAML file**. Due to the concise nature of the YAML format, a YAML file is recommended when your deployment includes only container instances.
- Container groups can share an external-facing IP address, one or more ports on the IP address, and a DNS label with an FQDN.
    - **External client access**. You must expose the port on the IP address and from the container to enable external clients to reach a container in your group.
    - **Port mapping**. Port mapping isn't supported because containers in a group share a port namespace.
    - **Deleted groups**. When a container group is deleted, its IP address and FQDN are released.

Configuration example

Consider the following example of a multi-container group with two containers.



The multi-container group has the following characteristics and configuration:

- The container group is scheduled on a single host machine and is assigned a DNS name label.
- The container group exposes a single public IP address with one exposed port.
- One container in the group listens on port 80. The other container listens on port 1433.
- The group includes two Azure Files file shares as volume mounts. Each container in the group mounts one of the file shares locally.

Things to consider when using container groups

Multi-container groups are useful when you want to divide a single functional task into a few container images. Different teams can deliver the images, and the images can have separate resource requirements.

Consider the following scenarios for working with multi-container groups. Think about what options can support your internal apps for the online retailer.

- **Consider web app updates**. Support updates to your web apps by implementing a multi-container group. One container in the group serves the web app and another container pulls the latest content from source control.
- **Consider log data collection**. Use a multi-container group to capture logging and metrics data about your app. Your application container outputs logs and metrics. A logging container collects the output data and writes the data to long-term storage.
- **Consider app monitoring**. Enable monitoring for your app with a multi-container group. A monitoring container periodically makes a request to your application container to ensure your app is running and responding correctly. The monitoring container raises an alert if it identifies possible issues with your app.
- **Consider front-end and back-end support**. Create a multi-container group to hold your front-end container and back-end container. The front-end container can serve a web app. The back-end container can run a service to retrieve data.

Azure Container Apps

There are many options for teams to build and deploy cloud native and containerized applications on Azure. Let's understand which scenarios and use cases are best suited for Azure Container Apps and how it compares to other container options on Azure.

Things to know about Azure Container Apps

Azure Container Apps is a serverless platform that allows you to maintain less infrastructure and save costs while running containerized applications. Instead of worrying about server configuration, container orchestration, and deployment details, Container Apps provides all the up-to-date server resources required to keep your applications stable and secure.

Common uses of Azure Container Apps include:

- Deploying API endpoints
- Hosting background processing jobs
- Handling event-driven processing
- Running microservices

Applications built on Azure Container Apps can dynamically scale based on the following characteristics:

- HTTP traffic
- Event-driven processing
- CPU or memory load
- Any KEDA-supported scaler

Things to consider when using Azure Container Apps

Azure Container Apps enables you to build serverless microservices and jobs based on containers. Distinctive features of Container Apps include:

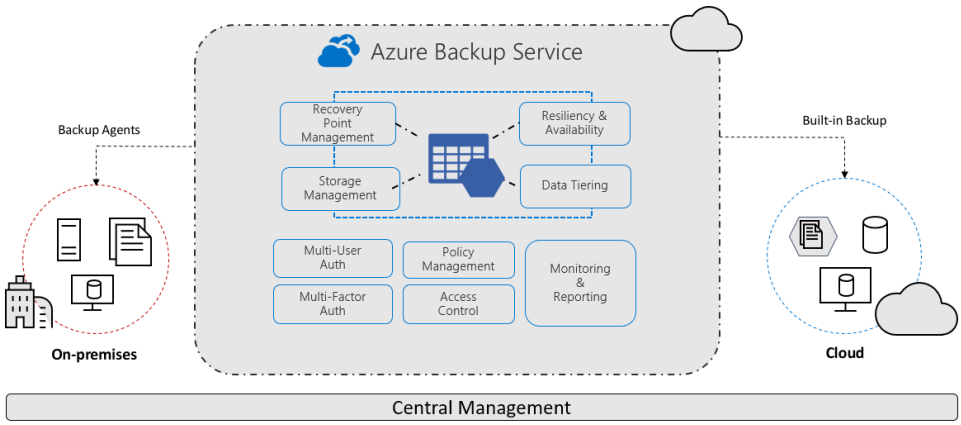
- Optimized for running general purpose containers, especially for applications that span many microservices deployed in containers.
- Powered by Kubernetes and open-source technologies like Dapr, KEDA, and envoy.

- Supports Kubernetes-style apps and microservices with features like service discovery and traffic splitting.
- Enables event-driven application architectures by supporting scale based on traffic and pulling from event sources like queues, including scale to zero.
- Supports running on demand, scheduled, and event-driven jobs.

Azure Container Apps doesn't provide direct access to the underlying Kubernetes APIs. If you would like to build Kubernetes-style applications and don't require direct access to all the native Kubernetes APIs and cluster management, Container Apps provides a fully managed experience based on best-practices. For these reasons, many teams may prefer to start building container microservices with Azure Container Apps.

What is Azure Backup?

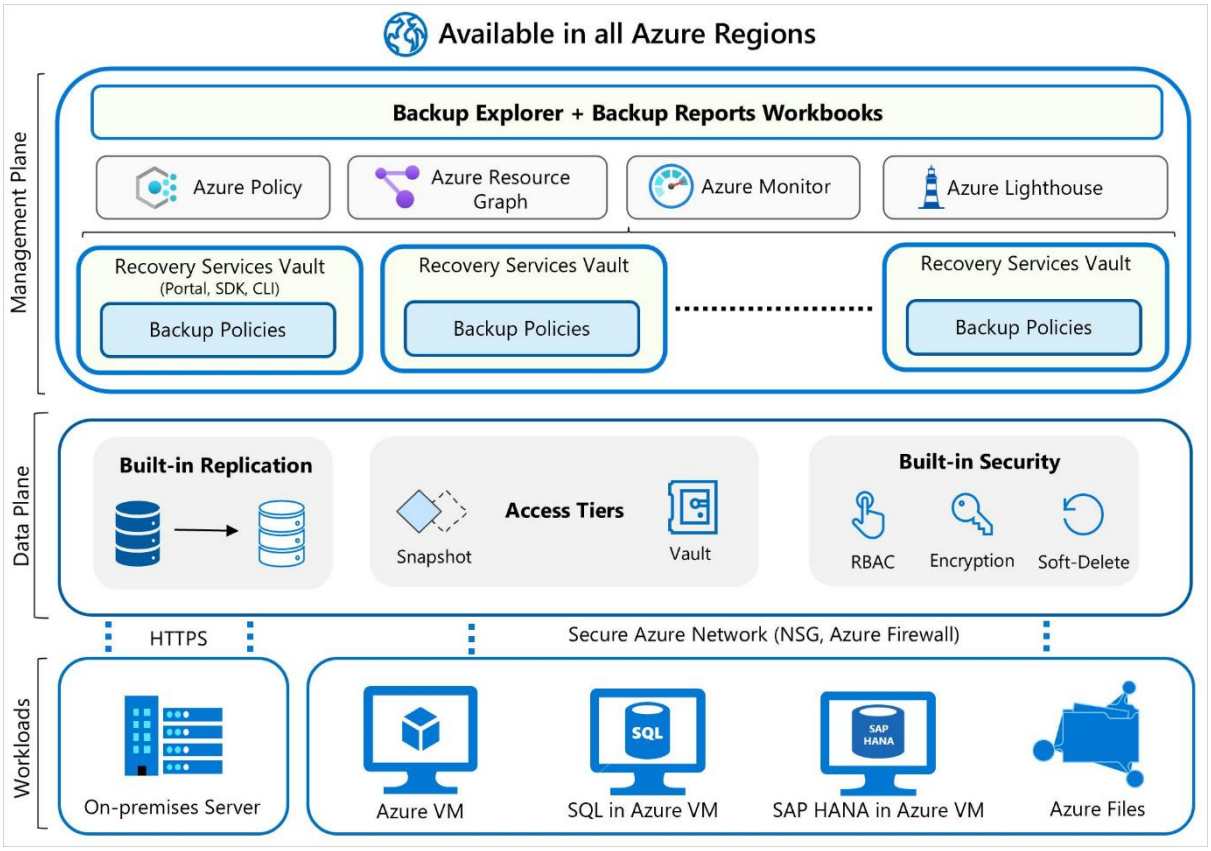
The Azure Backup service provides simple, secure, and cost-effective solutions to back up your data and recover it from the Microsoft Azure cloud.



Azure Backup definition

Azure Backup is an Azure service that provides cost effective, secure, and zero-infrastructure backup solutions for all Azure-managed data assets.

The centralized management interface makes it easy to define backup policies and protect a wide range of enterprise workloads, including Azure Virtual Machines, Azure Disks, SQL and SAP databases, Azure file shares, and blobs.



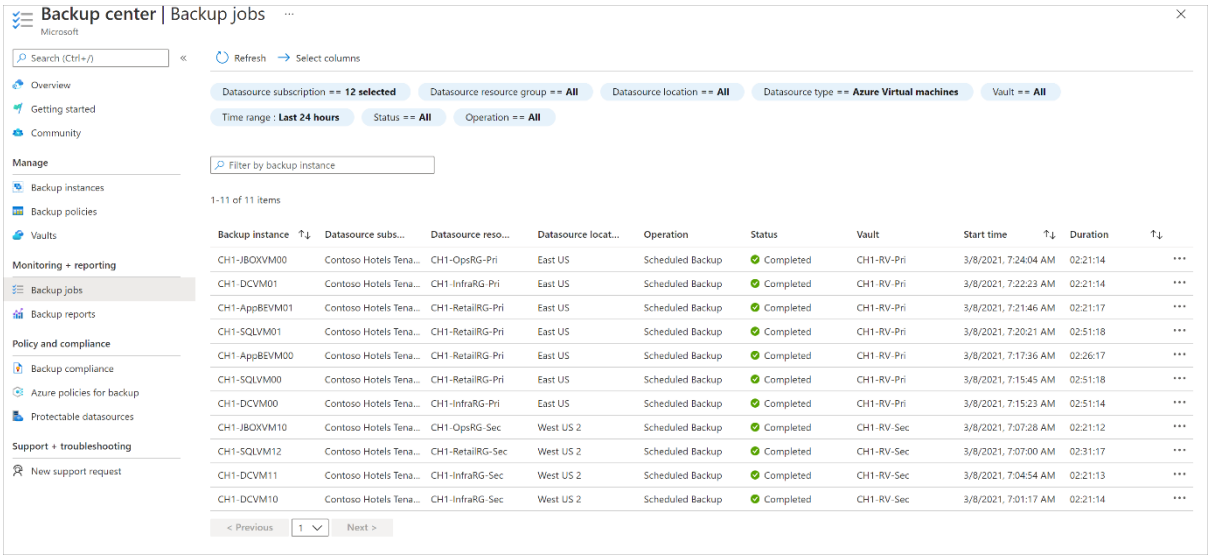
When to use Azure Backup

As the IT admin of your organization, you're responsible for meeting the compliance needs for all the data assets of the firm, and backup is a critical aspect. There are also various application admins in your company who need to do

self-service backup and restore to take care of issues like data corruption or rogue-admin scenarios. You're looking for an enterprise-class backup solution to protect all your workloads and manage them from a central place.

Azure Backup can provide backup services for the following data assets:

- On-premises files, folders, and system state
- Azure Virtual Machines (VMs)
- Azure Managed Disks
- Azure Files Shares
- SQL Server in Azure VMs
- SAP HANA (High-performance Analytic Appliance) databases in Azure VMs
- Azure Database for PostgreSQL servers
- Azure Blobs
- Azure Database for PostgreSQL - Flexible servers
- Azure Database for MySQL - Flexible servers
- Azure Kubernetes cluster



Key features

Let's look at some key features of Azure Backup.

Feature	Description	Usage
Zero-infrastructure backup solution	Unlike conventional backup solutions, no backup server or infrastructure is needed. Similarly, no backup storage needs to be deployed, because Azure Backup automatically manages and scales it.	Zero-infrastructure solution eliminates capital expenses and reduces operational expenses. It increases ease of use by automating storage management.
At-scale management	Natively manage your entire backup estate from a central console called Backup Center. Use APIs, PowerShell, and Azure CLI to automate Backup policy and security configurations.	Backup center simplifies data protection management at-scale by enabling you to discover, govern, monitor, operate, and optimize backup management from one unified console, which helps you to drive operational efficiency with Azure.
Security	Azure Backup provides built-in security to your backup environment, both when your data is in transit and at rest by using capabilities encryption, private endpoints, alerts, and so on.	Your backups are automatically secured against ransomware, malicious admins, and accidental deletions.

How do Recovery Time Objective and Recovery Point Objective work?

Recovery Time Objective (RTO) is the target time within which a business process must be restored after a disaster occurs to avoid unacceptable consequences. For instance, if a critical application goes down due to a server failure and the business can only tolerate a maximum of four hours of downtime, then the RTO is four hours.

Recovery Point Objective (RPO) is the maximum amount of data loss, measured in time, that your organization can sustain during an event.

How Azure Backup works

The simplest explanation of Azure Backup is that it backs up data, machine state, and workloads running on on-premises machines and VM instances to the Azure cloud. Azure Backup stores the backed-up data in Recovery Services vaults and Backup vaults.

For on-premises Windows machines, you can back up directly to Azure with the Azure Backup Microsoft Azure Recovery Services (MARS) agent. Alternatively, you can back up these Windows machines to a backup server, perhaps a System Center Data Protection Manager (DPM) or Microsoft Azure Backup Server (MABS). You can then back that server up to a Recovery Services vault in Azure.

If you're using Azure VMs, you can back them up directly. Azure Backup installs a backup extension to the Azure VM agent that's running on the VM, which allows you to back up the entire VM. If you only want to back up the files and folders on the VM, you can do so by running the MARS agent.

Azure Backup stores backed-up data in vaults: Recovery Services vaults and Backup vaults. A vault is an online-storage entity in Azure that's used to hold data such as backup copies, recovery points, and backup policies.

Supported backup types

Azure Backup supports full backups and incremental backups. Your initial backup is a full backup. The incremental backup is used by DPM/MABS use the incremental backup for disk backups, and used in all backups to Azure. As the name suggests, incremental backups only focus on the blocks of data that changed since the previous backup.

Azure Backup also supports SQL Server backup types. The following table outlines the support for SQL Server type backups:

Type	Description	Usage
Full	A full database backup backs up the entire database. It contains all the data in a specific database or in a set of filegroups or files. A full backup also contains enough logs to recover that data.	At most, you can trigger one full backup per day. You can choose to make a full backup on a daily or weekly interval.
Differential	A differential backup is based on the most recent full-data backup. It captures only the data that changed since the full backup.	At most, you can trigger one differential backup per day. You can't configure a full backup and a differential backup on the same day.
Multiple backups per day	Back up Azure VMs hourly with a minimum recovery point objective (RPO) of 4 hours and a maximum of 24 hours.	You can use Enhanced backup policy to set the backup schedule to 4, 6, 8, 12, and 24 hours (respectively) for new Azure offerings, such as Trusted Launch VM.
Selective disk backup	Selectively back up a subset of the data disks that are attached to your VM, then restore a subset of the disks that are available in a recovery point, both from instant restore and vault tier. Selective disk backup helps you manage critical data in a subset of the VM disks and use database backup solutions when you want to back up only their OS disk to reduce cost.	Azure Backup provides Selective Disk backup and restore capability using Enhanced backup policy.
Transaction Log	A log backup enables point-in-time restoration up to a specific second.	At most, you can configure transactional log backups every 15 minutes.

Azure Backup features and scenarios

What is Azure Backup?

Azure Backup is a built-in Azure service that provides secure backup for all Azure-managed data assets. It uses zero-infrastructure solutions to enable self-service backups and restores, with at-scale management at a lower and predictable cost. Azure Backup currently offers specialized backup solutions for Azure and on-premises virtual machines (VMs). Azure Backup also gives workloads like SQL Server or SAP High-performance Analytic Appliance (HANA) running in Azure VMs enterprise-class backup and restore options.

Azure Backup versus Azure Site Recovery

Both Azure Backup and Azure Site Recovery aim to make the system more resilient to faults and failures, but they use two different approaches. The primary goal of Backup is to maintain copies of stateful data that allow you to go back in time. Site Recovery, however, replicates the data in almost real time and allows for a failover.

In that sense, if there are issues like network or power outages, you can use availability zones. For a region-wide disaster (such as natural disasters), Site Recovery is used. Backups are used in cases of accidental data loss, data corruption, or ransomware attacks.

Additionally, the choice of a recovery approach depends on the criticality of the application, recovery point objective (RPO) and recovery time objective (RTO) requirements, and the cost implications.

Why use Azure Backup?

Traditional backup solutions, such as disk and tape, don't offer the highest level of integration with cloud-based solutions. Azure Backup has several benefits over more traditional backup solutions:

Zero-infrastructure backup: Azure Backup eliminates the need to deploy and manage any backup infrastructure or storage. There's no overhead in maintaining backup servers or scaling the storage up or down as the needs vary.

Long-term retention: Meet rigorous compliance and audit needs by retaining backups for many years, after which the built-in lifecycle management capability prunes the recovery points automatically.

Security: Azure Backup provides security to your backup environment, both when your data is in transit and at rest:

- **Azure role-based access control**: Role-based access control allows you to segregate duties within your team and grant only the amount of access to users necessary to do their jobs.
- **Encryption of backups**: Backup data is automatically encrypted using Microsoft-managed keys. Alternatively, you can encrypt your backed-up data using customer-managed keys stored in the Azure Key Vault.
- **No internet connectivity required**: When you use Azure VMs, all the data transfer happens only on the Azure backbone network without needing to access your virtual network. So no access to any IPs or fully qualified domain names (FQDNs) is required.
- **Soft delete**: With soft delete, the backup data is retained for 14 more days even after the deletion of the backup item. This retention protects against accidental deletion or malicious deletion scenarios, allowing the recovery of those backups with no data loss. Azure Backup also provides **Enhanced soft delete** that enables you to retain a deleted item in the *soft deleted* state for a longer duration.

Azure Backup also offers the ability to back up VMs encrypted with Azure Disk Encryption.

High availability: Azure Backup offers three types of replications:

- **Locally redundant storage (LRS)**: The lowest-cost option with basic protection against server rack and drive failures. We recommend it for noncritical scenarios.

- **Geo-redundant storage (GRS)**: The intermediate option has failover capabilities in a secondary region. We recommend it for backup scenarios.
- **Zone-redundant storage (ZRS)**: This option protects against datacenter-level failures by replicating your storage account synchronously across three Azure availability zones. We recommend it for high-availability scenarios.

Centralized monitoring and management: Azure Backup provides built-in monitoring and alerting capabilities in a Recovery Services vault. These capabilities are available without any other management infrastructure.

Azure Backup supported scenarios

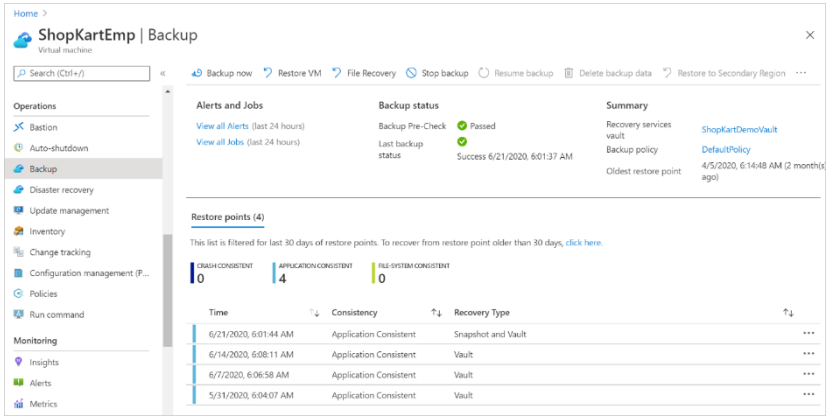
Azure Backup supports the following scenarios:

- **Azure VMs** - Back up Windows or Linux Azure VMs
  Azure Backup provides independent and isolated backups to guard against unintended destruction of the data on your VMs. Backups are stored in a Recovery Services vault with built-in management of recovery points. Configuration and scaling are simple, backups are optimized, and you can easily restore as needed.
- **On-premises** - Back up files, folders, and system state using the Microsoft Azure Recovery Services (MARS) agent. Or use Microsoft Azure Backup Server (MABS) or Data Protection Manager (DPM) server to protect on-premises VMs (Hyper-V and VMware) and other on-premises workloads.
- **Azure Files shares** - Azure Files provides snapshot management by Azure Backup.
- **SQL Server in Azure VMs** and **SAP HANA databases in Azure VMs** - Azure Backup offers stream-based, specialized solutions to back up SQL Server, or SAP HANA running in Azure VMs. These solutions take workload-aware backups that support different backup types such as full, differential and log, 15-minute RPO, and point-in-time recovery.

Back up an Azure virtual machine by using Azure Backup

Azure VMs are backed up by taking snapshots of the underlying disks at user-defined intervals and transferring those snapshots to the Recovery Services vault as per the customer-defined policy.

Recovery Services vault



Azure Backup uses a Recovery Services vault to manage and store the backup data. A vault is a storage-management entity, which provides a simple experience to carry out and monitor backup and restore operations. With Azure Backup, you don't need to worry about deploying or managing storage accounts. In fact, all you need to specify is the vault that you want to back up the virtual machine (VM) to. The backup data is transferred to the Azure Backup storage accounts (in a separate fault domain) in the background. The vault also acts as a role-based access control boundary to allow secure access to the data.

Snapshots

A snapshot is a point-in-time backup of all disks on the VM. For Azure VMs, Azure Backup uses different extensions for each supporting operating system:

Extension	OS	Description
VM Snapshot	Windows	The extension works with Volume Shadow Copy Service (VSS) to take a copy of the data on disk and in memory.
VM SnapshotLinux	Linux	The snapshot is a copy of the disk.

Depending on how the snapshot is taken and what it includes, you can achieve different levels of consistency:

- **Application consistent**
    - The snapshot captures the VM as a whole. It uses VSS writers to capture the content of the machine memory and any pending I/O operations.
    - For Linux machines, you need to write custom pre or post scripts per app to capture the application state.
    - You can get complete consistency for the VM and all running applications.
- **File system consistent**
    - If VSS fails on Windows, or the pre and post scripts fail on Linux, Azure Backup still creates a file-system-consistent snapshot.
    - During a recovery, no corruption occurs within the machine. But installed applications need to do their own cleanup during startup to become consistent.
- **Crash consistent**
    - This level of consistency typically occurs if the VM is shut down at the time of the backup.
    - No I/O operations or memory contents are captured during this type of backup. This method doesn't guarantee data consistency for the OS or app.

Backup policy

You can define the backup frequency and retention duration for your backups. Currently, the VM backup can be triggered daily or weekly, and can be stored for multiple years. The backup policy supports two access tiers: *snapshot tier* and the *vault tier*. By using the Enhanced policy, you can trigger hourly backups.

Selective disk backup: Azure Backup provides **Selective Disk backup and restore** capability using **Enhanced policy**. By using this capability, you can selectively back up a subset of the data disks that are attached to your VM. Then, you can restore a subset of the disks that are available in a recovery point, both from instant restore and vault tier. It helps you manage critical data in a subset of the VM disks and use database backup solutions when you want to back up only their OS disk to reduce cost.

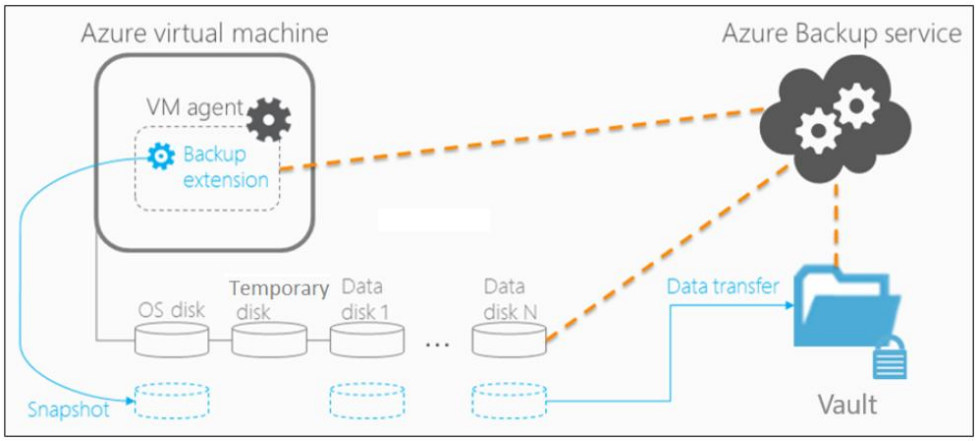
Snapshot tier: All the snapshots are stored locally for a maximum period of five days, in what is called the snapshot tier. For all types of operation recoveries, we recommended that you restore from the snapshots because it's faster to do so. This capability is called **instant restore**.

Vault tier: All snapshots are additionally transferred to the vault for more security and longer retention. At this point, the recovery point type changes to "snapshot and vault."

Backup process for an Azure virtual machine

Here's how Azure Backup completes a backup for Azure VMs:

1. For Azure VMs that are selected for backup, Azure Backup starts a backup job according to the backup frequency you specify in the backup policy.
2. During the first backup, a backup extension is installed on the VM, if the VM is running:
    - For Windows VMs, the VM Snapshot extension is installed.
    - For Linux VMs, the VM SnapshotLinux extension is installed.
3. After the snapshot is taken, the data is stored locally and transferred to the vault.
    - The backup is optimized by backing up each VM disk in parallel.
    - For each disk that's being backed up, Azure Backup reads the blocks on the disk and identifies and transfers only the data blocks that changed (the delta) since the previous backup.
    - Snapshot data might not be immediately copied to the vault. It might take several hours at peak times. Total backup time for a VM is less than 24 hours for daily backup policies.



You can additionally enable vault encryption with customer-managed keys (CMK). By using **Enhanced soft delete** for a Recovery Services vault, you can protect backups from deletion. You can also keep Enhanced soft delete *always on* to prevent turning it off, thus protecting your backups from accidental deletion or from malware attacks.

Restore virtual machine data

Azure Backup provides many ways to restore a VM. As explained earlier, you can either instantly restore from the snapshot tier (optimal for operational recoveries) or from the vault tier.

Restore option	Details
Create a new VM	Quickly creates and gets a basic VM up and running from a restore point. The new VM must be created in the same region as the source VM.
Restore disk	Restores a VM disk, which can then be used to create a new VM. The disks are copied to the resource group you specify. Azure Backup provides a template to help you customize and create a VM. Alternatively, you can attach the disk to an existing VM, or create a new VM. This option is useful if you want to customize the VM, add configuration settings that weren't there at the time of backup. Or, add settings that must be configured using the template or PowerShell.
Replace existing	You can restore a disk and use it to replace a disk on the existing VM. Azure Backup takes a snapshot of the existing VM before replacing the disk and stores it in the staging location you specify. Existing disks connected to the VM are replaced with the selected restore point. The current VM must exist. You can't use this option if the VM is deleted.
Cross region (secondary region)	Cross region restore can be used to restore Azure VMs in the secondary region, which is an Azure paired region. This feature is available for the following options: • Create a VM • Restore Disks We don't currently support the Replace existing disks option.

Introduction to monitoring

Monitoring tracks the state, health, behaviour, and performance of your applications and IT environment. One goal of monitoring is to ensure that your applications and environment are operating optimally, securely, and reliably. Another goal is to detect and help address any issues.

Monitoring includes the following key activities:

- **Data collection**: Metrics, logs, and log traces to provide insights into the functioning and performance of monitored components.
- **Data analysis**: Understand current state, predict potential issues, identify patterns, trends, and anomalies.
- **Alerts**: Trigger when specific conditions are met, such as high CPU usage or low disk space. Helps notify administrators or trigger automated responses.
- **Visualizations**: Present collected data in user-friendly visual formats to help administrators quickly assess system and resource status.
- **Diagnostics and troubleshooting**: Help identify the root causes of problems and make informed decisions to address them.

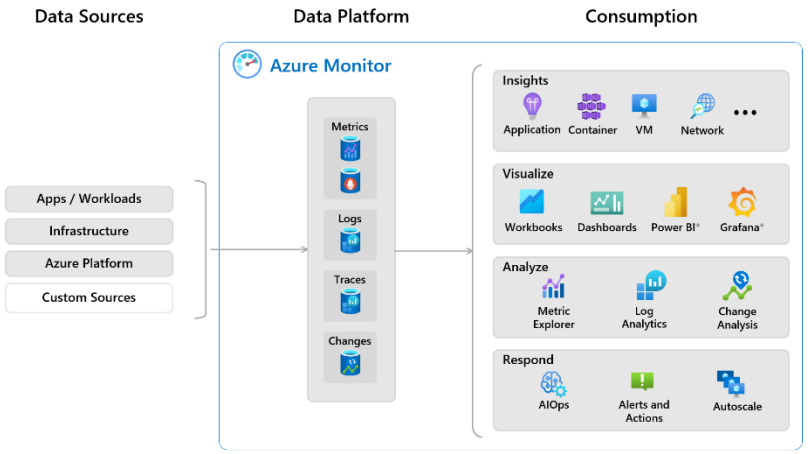
Monitoring provides the following important benefits:

- **Performance and cost optimization**: Identifies performance bottlenecks and areas for improving resource utilization, efficiency, and costs.
- **Proactive management**: Lets you take proactive rather than reactive measures to prevent downtime, disruptions, and other problems.
- **Reliability**: Provides quick identification, troubleshooting, and recovery when problems occur.
- **Capacity planning**: Helps you analyze historical usage patterns to aid in forecasting, planning, and infrastructure scaling.
- **Security monitoring**: Detects and responds to security threats, breaches, and suspicious activities to help maintain your system's security posture.
- **Compliance and governance monitoring**: Can monitor adherence to standards, regulations, and policies.

Overview of Azure Monitor

When you run applications built on various services and resources, a key element of monitoring is the ability to relate your applications' performance and health to the components they're built on. This *observability* lets you analyze and troubleshoot application issues effectively.

Azure Monitor provides features and tools for collecting, managing, and analyzing IT data from all of your Azure, other cloud, and on-premises resources. The following diagram shows a high-level architectural view of Azure Monitor.



Data collection and storage

As soon as you add resources to an Azure subscription, Azure Monitor starts collecting data about the resources. Azure Monitor provides the following capabilities for collecting, storing, and managing monitoring data:

- Native monitoring of your entire Azure deployment.
- Tools such as data collection agents and APIs for monitoring all layers in your stack. This includes applications and infrastructure, in and outside of Azure.
- Integration with the Azure Event Hubs data streaming service.
- Data transformations during ingestion to let you filter out data you don't need.
- Configurable data retention periods, archiving, and restore options.
- Pricing tier discounts based on data volume.
- A low-cost Basic logs plan for collecting and storing high-volume verbose logs you use for debugging, troubleshooting, and auditing. However, these aren't for analytics and alerts.

Data analysis and response

Azure Monitor offers a broad set of tools and capabilities to help you analyze and gain insights from your monitoring data. Azure Monitor includes the following features to support data analysis and response:

- An easy-to-use portal UI that lets you view, filter, and manipulate monitoring data.
- Kusto Query Language (KQL), a powerful query language that's optimized for ad-hoc queries, data exploration, and near-real time analysis of large volumes of data streamed from multiple sources.
- A range of tools for customizing your analysis, visualizations, alerting, and responses.
- Out-of-the-box recommended alerts.
- Ready-to-use monitoring experiences with advanced, built-in analysis and visualizations of your deployment.
- Autoscale to automatically add and remove resources according to the load on your application.
- Native machine learning and artificial intelligence capabilities that help you detect and respond to anomalies.

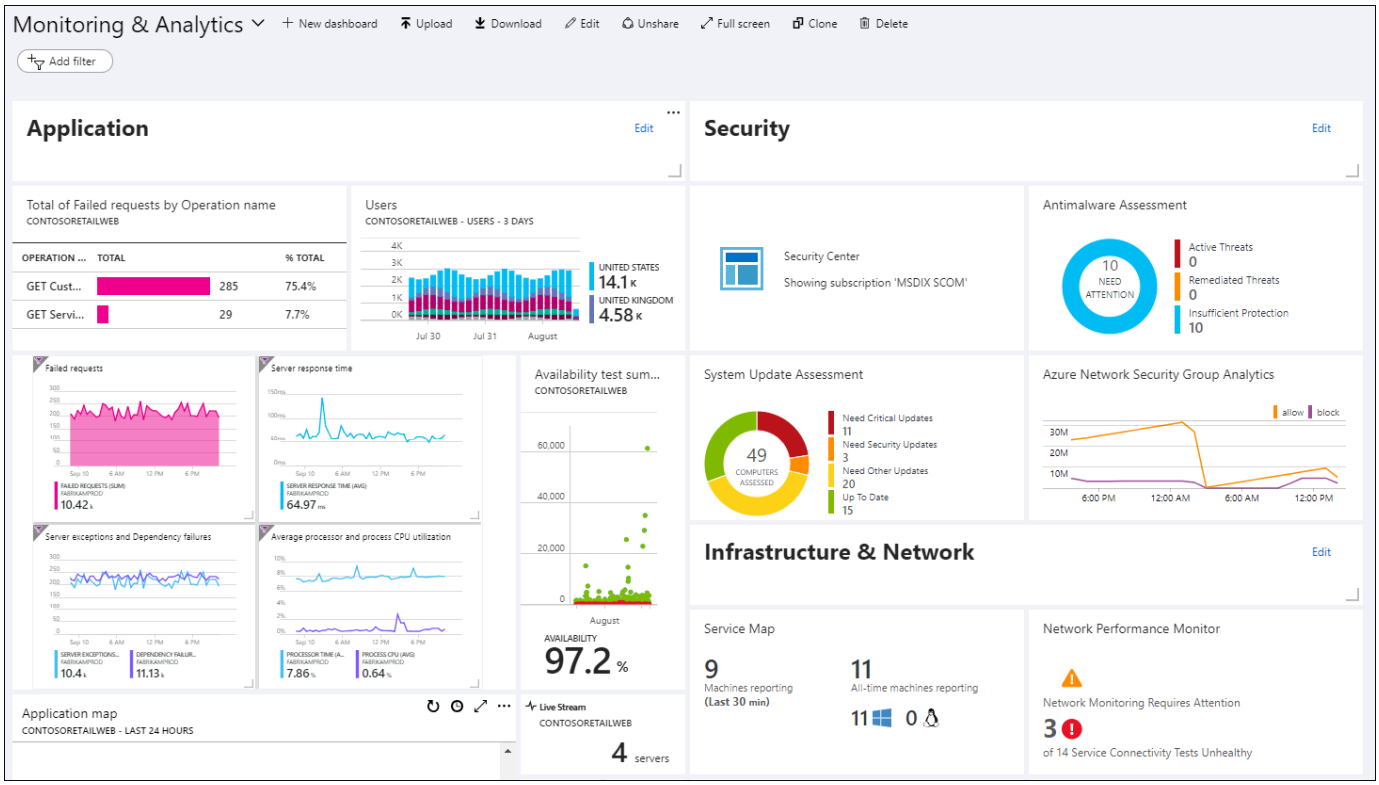
Alerts, workbooks, and visualizations

Interactive monitoring is one way to monitor your application. Another option is to configure alerts to send text messages or email to a person or team for further investigation. You can also trigger response actions in certain situations.

Azure Monitor workbooks provide a flexible canvas for analyzing data and creating rich visual reports in the Azure portal. Workbooks can tap into multiple Azure data sources and combine them into unified interactive experiences. You can use the ready-made workbooks that Azure Monitor provides, or create your own workbooks from predefined templates.

The following image shows three types of workbooks that display logged data in different chart and table formats.

You can add the visualizations you create in Azure Monitor to Azure dashboards, which let you combine different kinds of data into a single pane in the Azure portal.



Metrics

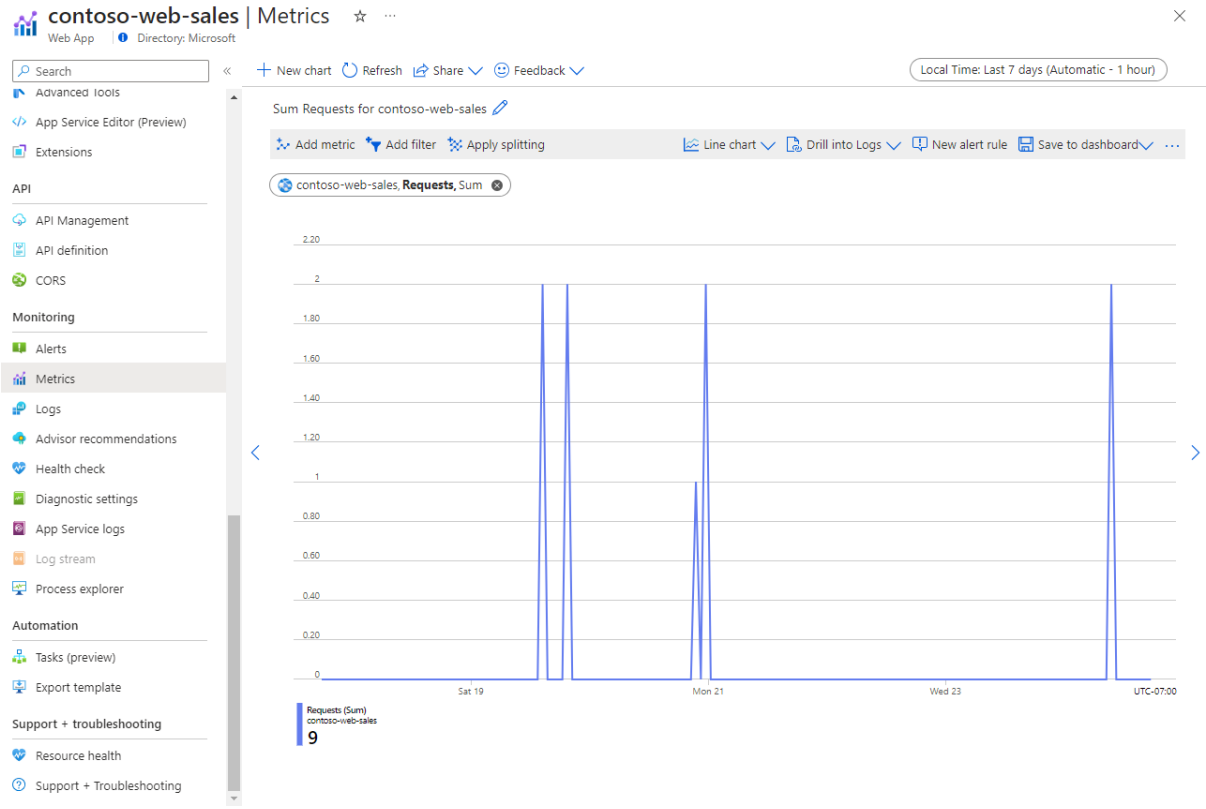
Metrics are quantitative measurements that show snapshots of application or resource performance. Metrics are typically numeric values that you can measure over time.

Metrics can provide you with an understanding of various aspects of an application or resource, such as resource utilization, response times, error rates, and throughput. Common examples of metrics include CPU usage, memory usage, network latency, and transaction rates.

A list of resource-specific metrics is automatically available for each resource type in your Azure subscription. You can use the Azure Monitor Metrics Explorer to interactively analyze the data in your metrics database and chart the values of multiple metrics over time.

To see the metrics for any resource in the Azure portal, select **Metrics** under **Monitoring** in the left navigation on that resource's page. Then select the metric you need from the **Metric** dropdown. You can pin the charts to a dashboard to view them with other visualizations.

For example, the following **Requests** metric line chart shows the sum aggregation of requests for the **Contoso-web-sales** application.



Azure Monitor can collect several types of metrics, including:

- **Azure platform metrics**: Azure Monitor starts collecting metrics data from Azure resources as soon as they're added to a subscription. A list of resource-specific metrics is automatically available for each Azure resource type.
- **Custom metrics**: Azure Monitor can also collect metrics from other sources, including applications and agents running on VMs. You can send custom metrics to Azure Monitor via the Azure Monitor Agent, other agents and extensions, or directly to the Azure Monitor REST API.
- **Prometheus metrics**: Azure Monitor managed service for Prometheus collects metrics from Azure Kubernetes Service (AKS) or other Kubernetes clusters. Prometheus metrics share some characteristics with platform and

custom metrics, but have different features to support open-source analysis and alerting tools like PromQL and Grafana.

Metrics are well-suited for real-time monitoring. You can use metrics to trigger alerts when defined thresholds are reached.

Logs

Logs are textual records of events, actions, and messages that occur in a resource or application. While metrics are numeric, logs can include the following data:

- **Text**: Human-readable text entries that provide context, details, and descriptions of events.
- **Unstructured data**: Log entries in various formats that don't fit neatly into predefined numerical values.
- **Contextual information**: Insights into the context surrounding an event, which is invaluable for root cause analysis.

Logs can capture information about errors, warnings, user actions, and application state changes. Logs provide detailed narratives of events in a given context. This makes them crucial for troubleshooting, debugging, and understanding event sequences that lead to issues. Logs are essential for retrospective analysis of issues, helping to reconstruct the chain of events that led up to a problem.

Azure Monitor Logs is a feature of Azure Monitor that lets you store, manage, and analyze log and performance data from monitored resources. To collect and analyze all your data, you set up a common workspace called a Log Analytics workspace. You configure your resources to send their data to that workspace.

Once you configure the workspace and start logging data, you can use Azure Monitor Logs to explore and analyze the data. You can work with log queries and their results interactively in the Log Analytics user interface.

You can use log queries in the following scenarios:

- Use a basic query to answer a common question.
- Do complex data analysis to identify critical patterns in your monitoring data.
- Use queries in alert rules to be proactively notified of issues.
- Visualize query results in a workbook or dashboard.

Azure Monitor Insights, visualizations, and actions

You can also use alerts and automated actions to proactively respond to and sometimes correct application issues.

Insights

Some Azure resource providers have created visualizations that provide a customized monitoring experience and require minimal configuration. Insights are large, scalable, curated visualizations.

Azure Monitor includes many types of Insights. In the Azure portal, select **Insights Hub** in the Azure Monitor navigation to list and access all the available types of Insights.

The following sections describe some of the largest, most common Azure Monitor Insights.

Application Insights

The Application Insights feature of Azure Monitor provides application performance monitoring (APM) from app development, through test, and into production. You can *proactively* monitor to see how well an application is performing and *reactively* review application execution data to find the cause of an incident.

Along with collecting metrics and telemetry data that describe application activities and health, you can use Application Insights to collect and store application *trace logging* data. The log trace is associated with other telemetry to give a detailed view of activity. To add trace logging to existing applications, you only need to provide a destination for the logs. You seldom need to change the logging framework.

Application Insights supports *distributed tracing*, which is also known as distributed component correlation. This feature allows searching for and visualizing the end-to-end flow of a specific execution or transaction. The ability to trace activity from end to end is important for applications built as distributed components or microservices.

Application Insights also includes the following features:

- **Live metrics**: Observe activity from your deployed application in real time with no effect on the host environment.
- **Availability monitoring**: Also known as *synthetic transaction monitoring*, probes the external endpoints of your applications to test overall availability and responsiveness over time.
- **Usage monitoring**: Helps you understand which features are popular with users and how users interact and use your application.
- **Smart detection**: Detects failures and anomalies automatically through proactive telemetry analysis.
- **Application Map**: A high-level, top-down view of your application architecture with at-a-glance visual references to component health and responsiveness.

Container Insights

Container Insights gives you performance visibility into containerized workloads deployed to Azure Kubernetes Service (AKS) or Azure Container Instances. Container Insights collects container logs and metrics from controllers, nodes, and containers that are available through the Metrics API. After you enable monitoring from AKS clusters, these metrics and logs are automatically collected for you through a containerized version of the Log Analytics agent.

VM Insights

VM Insights monitors and analyzes the performance and health of your Azure Windows and Linux VMs, including VMs hosted on-premises or in another cloud. VM Insights identifies VM processes, application dependencies, and interconnected dependencies on external processes.

Network Insights

Network Insights provides a comprehensive visual representation of health and metrics for all deployed network resources through topologies, without requiring any configuration. Network Insights also provides access to network monitoring capabilities like Connection Monitor, flow logging for network security groups (NSGs), Traffic Analytics, and other diagnostic features.

Visualizations

Visualizations such as charts and tables are effective tools for summarizing monitoring data and presenting it to audiences. Azure Monitor has its own features for visualizing monitoring data, and uses other Azure services for publishing data to different audiences. Power BI and Grafana aren't officially part of Azure Monitor, but are core integrations to tell the monitoring story.

The following sections describe some Azure Monitor and external tools for visualizing and presenting monitoring data.

Workbooks

Workbooks provide a flexible canvas for analyzing data and creating rich visual reports in the Azure portal. Workbooks can query data from multiple data sources and combine and correlate data from multiple data sets in one visualization, giving you easy visual representation of your system. Workbooks are interactive, with data updating in real time, and can be shared across teams.

You can use the workbooks that Azure Monitor Insights provide, use the workbook template library, or create your own workbooks. In the Azure portal, select **Workbooks** in the Azure Monitor left navigation to see and access the available workbooks and templates.

Dashboards

Dashboards let you combine different kinds of data into a single pane in the Azure portal. You can add the output of any log query or metrics chart to an Azure dashboard, and optionally share the dashboard with other Azure users. For example, you could create a dashboard that shows a graph of metrics, a table of activity logs, and a usage chart from Application Insights.

Power BI

Power BI is a business analytics service that provides interactive visualizations across various data sources. You can configure Power BI to automatically import log data from Azure Monitor to take advantage of these visualizations. Power BI is an effective way to make data available to other people within and outside your organization.

Grafana

Grafana is an open platform for operational dashboards. Grafana includes the Azure Monitor data source plugin to visualize Azure Monitor metrics and logs. Azure Managed Grafana optimizes this experience for Azure-native data stores such as Azure Monitor and Azure Data Explorer.

Grafana also has popular plugins and dashboard templates for non-Microsoft application performance monitoring tools such as Dynatrace, New Relic, and AppDynamics. Grafana includes AWS CloudWatch and GCP BigQuery plugins for multicloud monitoring in a single pane of glass. You can use these resources to visualize Azure Monitor data alongside other metrics that these other tools collect.

Actions

An effective monitoring solution proactively responds to critical events without the need for an individual or team to notice the issue. The response could be a text or email to an administrator, or an automated process that attempts to correct an error condition.

Azure Monitor works with the following types of automated alerting and responses.

Artificial Intelligence for IT Operations (AIOps)

AIOps describes the application of artificial intelligence and machine learning techniques to enhance and automate aspects of IT operations and infrastructure management. Azure Monitor provides features that use machine learning and artificial intelligence to automate data-driven tasks, predict capacity usage, identify performance issues, and detect anomalies.

These features simplify IT monitoring and operations without requiring machine learning expertise. If you have machine learning expertise, you can apply more machine learning to the data Azure Monitor collects by using Azure Machine Learning services.

Azure Monitor alerts

Alerts notify you of critical conditions and can take corrective action. Alert rules can be based on metric or log data. Metric alert rules provide near-real time alerts based on collected metrics. Log alert rules based on log data allow for complex logic across data from multiple sources.

Alert rules use *action groups*, which can take actions such as sending email or SMS notifications. Action groups can send notifications using webhooks to trigger external processes or to integrate with IT service management tools. You can share action groups, actions, and sets of recipients across multiple rules.

Autoscale

Autoscale lets you dynamically adjust the number of resources running to handle the load on your applications. To save money or increase performance, you can create rules that use Azure Monitor metrics to determine when to automatically add or remove resources. You can specify a minimum and maximum number of instances and the logic for when to increase or decrease resources.

Explore the different alert types that Azure Monitor supports

Azure Monitor is a powerful reporting and analytics tool. You can use it for insights into the behavior and running of your environment and applications. You can then respond proactively to faults in your system.

After the downtime that your customers faced, you set up monitoring on your key resources in Azure. With the monitoring in place, you want to make sure the right people are being alerted at the right level.

In this unit, you learn how Azure Monitor receives resource data, what makes up an alert, and how and when to use an alert. Finally, you learn how to create and manage your own alerts.

Data types in Azure Monitor

Azure Monitor receives data from target resources like applications, operating systems, Azure resources, Azure subscriptions, and Azure tenants. The nature of the resource defines which data types are available. A data type can be a *metric*, a *log*, or both a metric and a log:

- The focus for *metric*-based data types is the numerical time-sensitive values that represent some aspect of the target resource.
- The focus for *log*-based data types is the querying of content data held in structured, record-based log files that are relevant to the target resource.

There are three signal types that you can use to monitor your environment:

- **Metric** alerts provide an alert trigger when a specified threshold is exceeded. For example, a metric alert can notify you when CPU usage is greater than 95 percent.
- **Activity log** alerts notify you when Azure resources change state. For example, an activity log alert can notify you when a resource is deleted.
- **Log** alerts are based on things written to log files. For example, a log search alert can notify you when a web server returns a particular number of 404 or 500 responses.

Composition of an alert rule

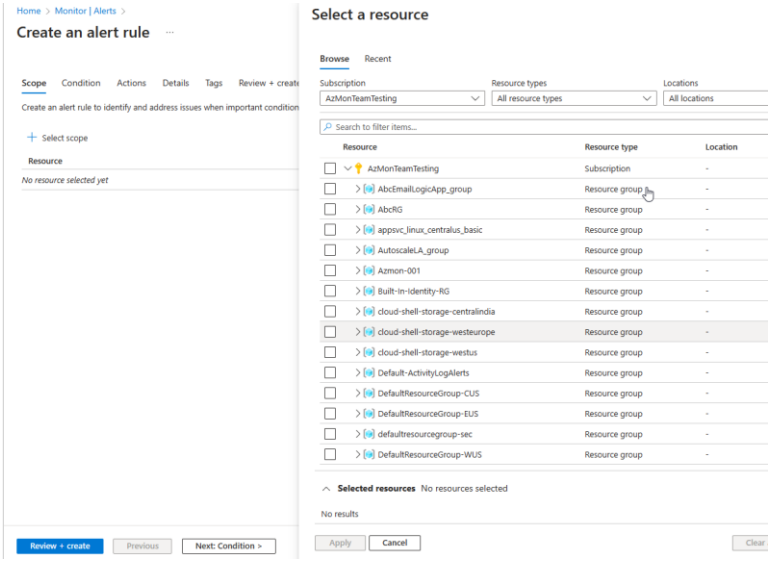
Every alert or notification available in Azure Monitor is the product of a rule. Some of these rules are built into the Azure platform. You can use alert rules to create custom alerts and notifications. No matter which target resource or data source you use, the composition of an alert rule remains the same.

- **RESOURCE**
    - The *target resource* for the alert rule. You can assign multiple target resources to a single alert rule. The type of resource defines the available signal types.
- **CONDITION**
    - The *signal type* used to assess the rule. The signal type can be a metric, an activity log, or logs. There are others, but this module doesn't cover them.

- The *alert logic* applied to the data that's being supplied via the signal type. The structure of the alert logic changes depending on the signal type.

- **ACTIONS**
  - The *action*, like sending an email, sending a Short Message Service (SMS) message, or using a webhook.
  - An *action group*, which typically contains a unique set of recipients for the action.

- **ALERT DETAILS**
  - An *alert name* and an *alert description* that specify the alert's purpose.
  - The *severity* of the alert if the criteria or logic test evaluates true. The five severity levels are:
    - **0**: Critical
    - **1**: Error
    - **2**: Warning
    - **3**: Informational
    - **4**: Verbose



Scope of alert rules

You can get monitoring data from across most of the Azure services and report on it by using the Azure Monitor pipeline. In the Azure Monitor pipeline, you can create alert rules for these items and more:

- Metric values
- Log search queries
- Activity log events
- Health of the underlying Azure platform
- Tests for website availability

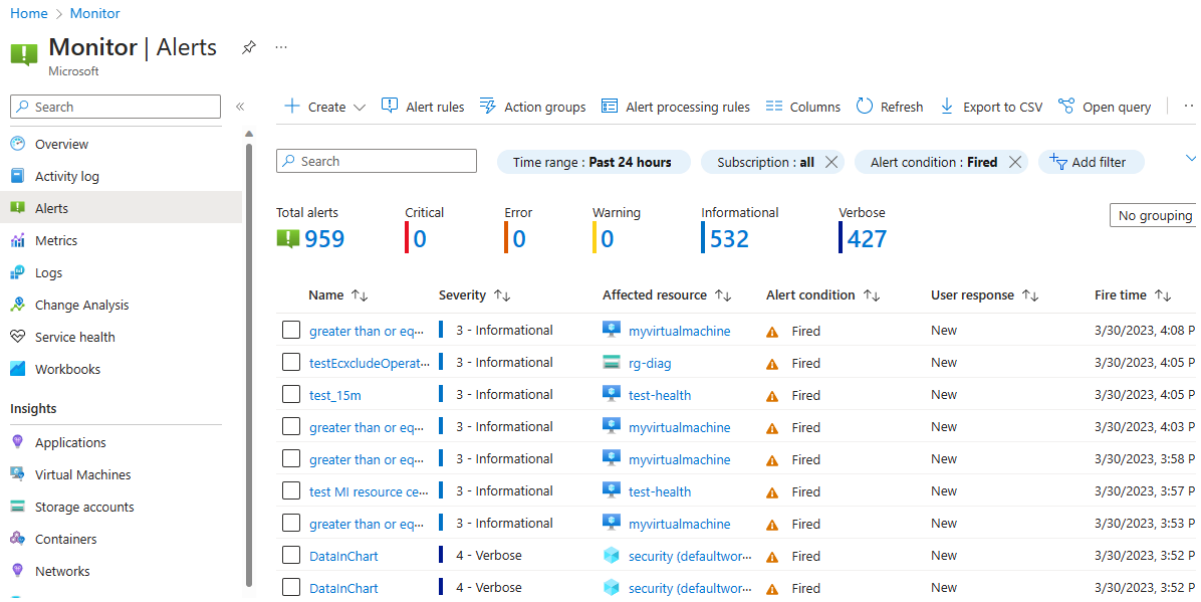
Manage alert rules

Not every alert rule that you create needs to run forever. With Azure Monitor, you can specify one or more alert rules and enable or disable them, as needed.

As an Azure solution architect, you want to use Azure Monitor to enable tightly focused and specific alerts before any application change. Then, disable the alerts after a successful deployment.

Alert summary view

The alert page shows a summary of all alerts. You can apply filters to the view by using one or more of the following categories: subscriptions, alert condition, severity, or time ranges. The view includes only alerts that match these criteria.



Alert condition

The system sets the alert condition.

- When an alert fires, the alert's monitor condition is set to **Fired**.
- After the underlying condition that caused the alert to fire clears, the monitor condition is set to **Resolved**.

Features of Azure Monitor logs

Azure Monitor is a service for collecting and analyzing telemetry. It helps you get maximum performance and availability for your cloud applications and for your on-premises resources and applications. It shows how your applications are performing and identifies any issues with them.

Data collection in Azure Monitor

Azure Monitor collects two fundamental types of data: metrics and logs. Metrics tell you how a resource is performing and the other resources that it's consuming. Logs contain records that show when resources are created or modified.

The following diagram gives a high-level view of Azure Monitor. On the left are the data-monitoring sources: Azure, operating systems, and custom sources. At the center of the diagram are the data stores for metrics and logs. On the right are the functions that Azure Monitor performs with this collected data, such as analysis, alerting, and streaming to external systems.

Azure Monitor collects data automatically from a range of components. For example:

- **Application data**: Data that relates to your custom application code.
- **Operating-system data**: Data from the Windows or Linux virtual machines that host your application.
- **Azure resource data**: Data that relates to the operations of an Azure resource, such as a web app or a load balancer.
- **Azure subscription data**: Data that relates to your subscription, including data about Azure health and availability.
- **Azure tenant data**: Data about your Azure organization-level services, such as Microsoft Entra ID.

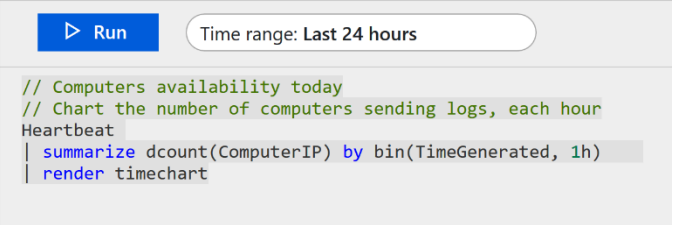
Because Azure Monitor is an automatic system, it begins to collect data from these sources as soon as you create Azure resources like virtual machines and web apps. You can extend the data that Azure Monitor collects by:

- **Enabling diagnostics**: For some resources, such as Azure SQL Databases, you'll receive full information about a resource only after you've enabled diagnostic logging for it. You can use the Azure portal, the Azure CLI, or PowerShell to enable diagnostics.
- **Adding an agent**: For virtual machines, you can install the Log Analytics agent and configure it to send data to a Log Analytics workspace. This agent increases the amount of information that's sent to Azure Monitor.

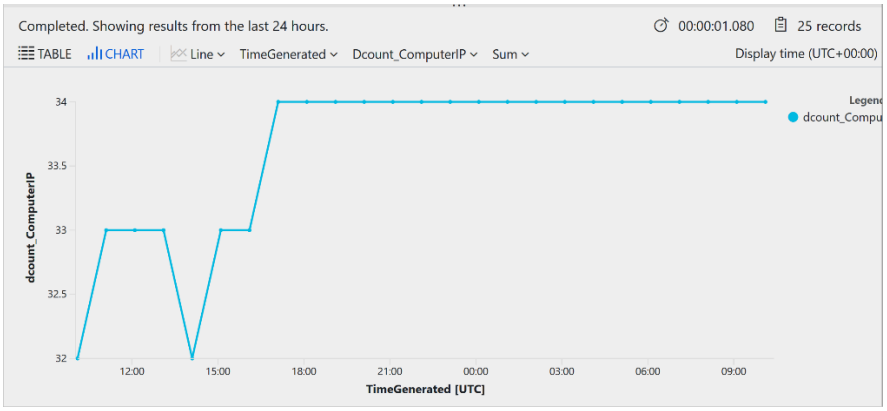
Your developers might also want to send data to Azure Monitor from custom code such as a web app, an Azure function, or a mobile app. They send data by calling the Data Collector API. You can communicate with this REST interface through HTTP. This interface is compatible with various development frameworks such as .NET Framework, Node.js, and Python. Developers can choose their favorite language and framework to log data in Azure Monitor.

Logs

Logs contain time-stamped information about resource changes. The type of information recorded varies by log source. The log data is organized into records, with different sets of properties for each type of record. The logs can include numeric values like Azure Monitor metrics, but most include text data rather than numeric values.

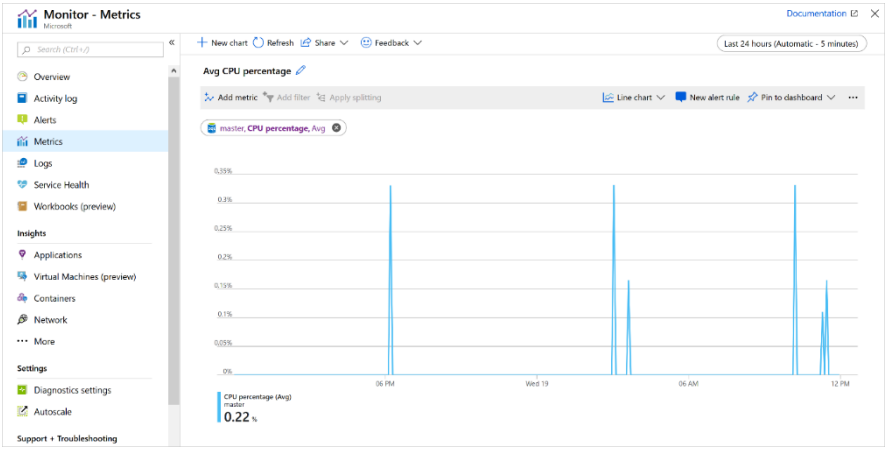


You can log data from Azure Monitor in a Log Analytics workspace. Azure provides an analysis engine and a rich query language. The logs show the context of any problems, and they're useful for identifying root causes.



Metrics

Metrics are numerical values that describe some aspect of a system at a point in time. Azure Monitor can capture metrics in near-real time. The metrics are collected at regular intervals, and they're useful for alerting because of their frequent sampling. You can use various algorithms to compare a metric to other metrics and observe trends over time.



Metrics are stored in a time-series database. This data store is most effective for analyzing time-stamped data. Metrics are suited for alerting and fast detection of issues. They can tell you about system performance. If needed, you can combine them with logs to identify the root cause of issues.

Analysing logs by using Kusto

To retrieve, consolidate, and analyse data, you can specify a query to run in Azure Monitor logs. You can write a log query with the Kusto query language, which Azure Data Explorer also uses.

You can test log queries in the Azure portal so you can work with them interactively. You'll typically start with basic queries, then progress to more advanced functions as your requirements become more complex.

In the Azure portal, you can create custom dashboards, which are targeted displays of resources and data. You can build each dashboard from a set of tiles. Each tile might show a set of resources, a chart, a data table, or some custom text. Azure Monitor provides tiles that you can add to dashboards; for example, you might use a tile to display a Kusto query's results in a dashboard.

In the example scenario, the operations team can consolidate its monitoring data by visualizing it in charts and tables. These tools are effective for summarizing data and presenting it to different audiences.

By using Azure dashboards, you can combine various kinds of data—including both logs and metrics—into a single pane in the Azure portal. For example, you might want to create a dashboard that combines tiles that show a graph of metrics, a table of activity logs, charts from Azure Monitor, and a log query's output.