SECURING AZURE ENVIRONMENTS WITH AZURE ACTIVE DIRECTORY

Architecture and Design Guide

Abstract

Identity guidance and best practices for designing and securing isolated environments in Azure.
For the latest version of this document please check: https://aka.ms/AzureADSecuredAzure.

Last updated: July 2020

AUTHORS

Ramiro Calderon, Principle Program Manager

Michael Duddington, Senior Program Manager

Barbara Winter Selden, Senior Program Manager

ACKNOWLEDGEMENTS

We had many internal contributors, and many internal and external reviewers.

We are especially grateful to our Azure Elite partners and Identity Advisors who reviewed this prior to publication.

Contents

About this paper

Organization's environments are designed to empower employees, partners, and contractors to be productive. Most organizations therefore initially onboard to Azure using their corporate production Azure Active Directory (Azure AD) tenant created with Office 365, Microsoft Dynamics, or another Microsoft online service. That way users and administrators may continue using their corporate user accounts for day-to-day work and management of Azure resources. However, it's imperative that productivity is balanced with security.

From a risk management perspective, you may want to consider separating Azure resources either within a tenant by using delegated administration, or in multiple tenants. While separation can increase overhead in terms of design, processes, and implementation, it can also segregate and mitigate the risks of bad actors, compromised credentials, and operational errors.

In Microsoft Azure, Azure AD is the identity governance and administration layer that is used to manage access to resources such as instances of virtual machines, databases, applications, APIs, websites, etc. This identity layer is the control plane that helps protect your resources from intruders.

In this paper, we describe the architectures and best practices for implementing identity and access management across separate Azure environments. Not all organizations need to run separate environments. This document will help you understand if this configuration is appropriate for your organization.

We begin with an Introduction to delegated administration and isolated environments. In this introduction we describe various deployment scenarios and critical considerations for deciding if separate environments are appropriate for your organization. Ultimately, we help you choose the right architecture for your organization: Delegated administration in a single tenant, Resource isolation in multiple tenants, or Resource and identity isolation in multiple tenants. We then provide a comprehensive list of design considerations, or best practices.

Important information on foundational Azure and Azure AD knowledge

Understanding that not everyone is deeply familiar with how Azure and Azure AD work, we have provided a detailed appendix on topics relevant to environment separation. *We strongly urge you to start with these topics if you have any questions about this foundational knowledge, or if you want a quick review before diving into this paper.*

- Azure Active Directory fundamentals details the terminology used in this document, as well as Azure AD functional areas such as authentication, authorization, administration, governance, and consumer identity management.
- Azure resource management fundamentals details the resource management model in Azure; how management groups, subscriptions, resource groups, and resources are related and accessed.
- Azure resource management with Azure AD details how the billing systems and roles for different types of agreements interact with privileged roles, how Azure AD roles and Azure role-based access control (RBAC) work together, how Conditional Access can help secure your environment, and information on managed and consumer identities.
- Identity considerations for Infrastructure as a Service details how Azure Active directory Domain Services relates to isolated environments.

Introduction to delegated administration and isolated environments

A single-tenant architecture with delegated administration is often adequate for separating environments. As detailed in other sections of this paper, Microsoft provides many tools to do this. However, there may be times when your organization requires a degree of isolation beyond what can be achieved in a single tenant.

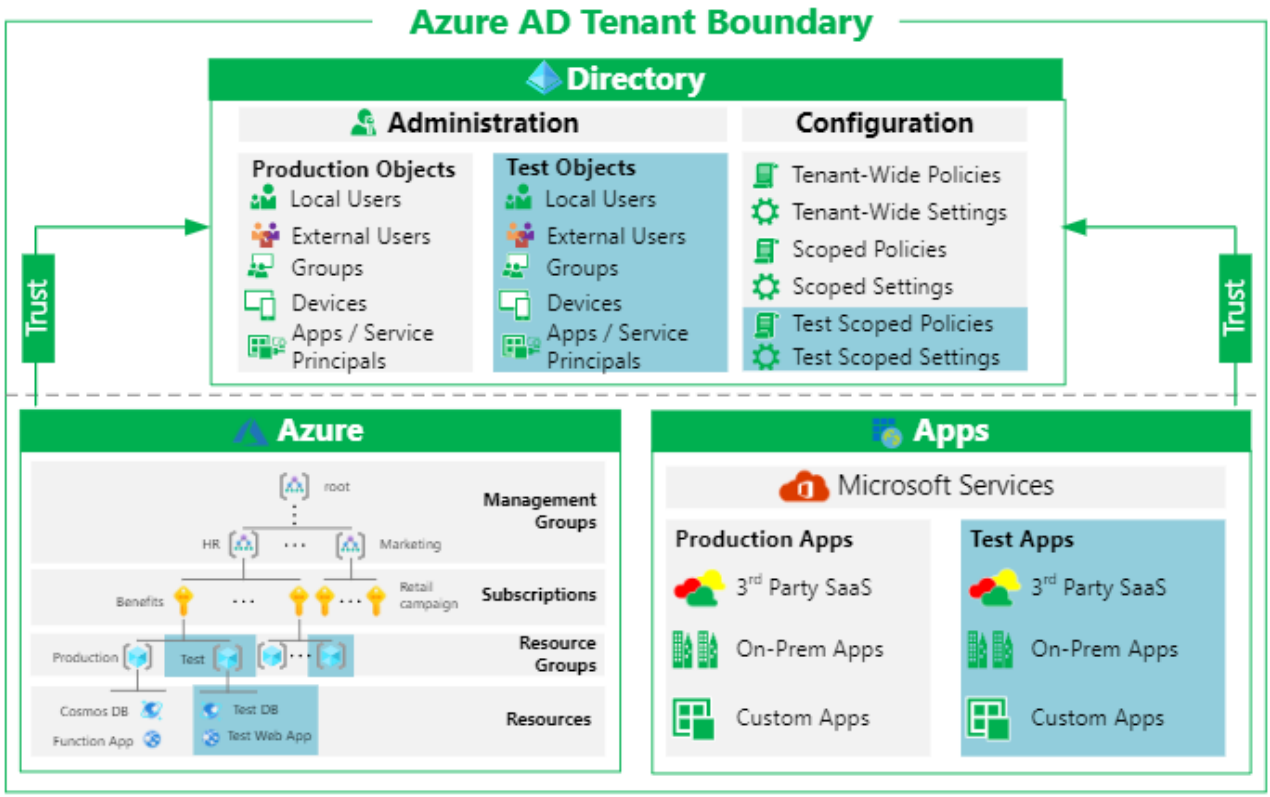
Before discussing specific architectures, it is important to understand:

- how a typical single tenant works.
- the relationships between Azure resources and Azure AD tenants.
- common requirements driving isolation.

Azure AD tenant as a security boundary

An Azure AD tenant provides identity and access management capabilities to applications and resources used by the organization. An identity is a directory object that can be authenticated and authorized for access to a resource. Identity objects exist for human identities, and non-human identities. Non-human entities include application objects, service principles, managed identities, and devices. The tenant is an identity security boundary that is under the control of global administrators. Within this security boundary, administration of subscriptions, management groups, and resource groups can be delegated to segment administrative control. While not directly interacting, these groupings are dependent on tenant-wide configurations of policies and settings. And those settings and configurations are under the control of the Azure AD Global Administrators.

In this section we identify how resources, configuration, and administration work together, and the common for reasons for segregating of environments in each area. For more detailed information on Azure resources and how Azure resources are controlled by Azure AD, see the appendix.



Resources in a tenant

Azure AD is used to grant objects representing identities access to applications and Azure resources. In that sense both Azure resources and applications trusting Azure AD are *resources that can be managed with Azure AD.* In this section, we will cover how this access to resources is modeled in Azure AD, and the mutual effects between resources and the Azure AD tenant.

Access to apps that use Azure AD

Identities can be granted access to many types of applications, including but not limited to:

- Microsoft productivity services such as Exchange Online, Microsoft Teams, and SharePoint Online
- Microsoft IT services such as Azure Sentinel, Microsoft Intune, and Microsoft Defender ATP.
- Microsoft Developer tools such as Azure DevOps
- SaaS solutions such as Salesforce and ServiceNow
- On-premises applications integrated with hybrid access capabilities such as Azure AD Application Proxy
- Custom in-house developed applications.

Applications that use Azure AD require directory objects to be configured and managed in the trusted Azure AD tenant. Examples of directory objects include application registrations, service principals, groups, and schema attribute extensions.

Access to Azure resources

Users, groups, and service principal objects in the Azure AD tenant are granted roles by using Role Based Access Control (RBAC). Azure resources, resource groups, subscriptions, and management groups are accessed through using these assigned RBAC roles.

Azure resources that support Managed Identities generate service principles that allow resources to authenticate, be granted access to, and be assigned roles to other resources within the Azure AD tenant boundary.

Applications using Azure AD for sign-in may also leverage Azure resources such as compute or storage as part of its implementation. For example, a custom application that runs in Azure and trusts Azure AD for authentication will have both directory objects and Azure resources.

Lastly, all Azure resources in the Azure AD tenant affect tenant-wide Azure Quotas and Limits.

Access to Directory Objects

As outlined in the diagram above, identities, resources, and their relationships are represented in an Azure AD tenant as directory objects. Examples of directory objects include users, groups, service principals, and app registrations.

Having a set of directory objects in the Azure AD tenant boundary engenders the following:

- **Visibility.** Identities can discover or enumerate resources**,** users, groups, access usage reporting and audit logs based on their permissions. For example, a member of the directory can discover users in the directory per Azure AD default user permissions.
- **Applications can affect objects.** Applications can manipulate directory objects through Microsoft Graph as part of their business logic. Typical examples include reading/setting user attributes, updating user's calendar, sending emails on behalf of the user, etc. Consent is necessary to allow applications to affect the tenant. Administrators can consent for all users**.** For more information, see Permissions and consent in the Microsoft identity platform.
    - Use caution when using application permissions. For example, with Exchange Online, you should scope application permissions to specific mailboxes and permissions.
- **Throttling and service limits.** Runtime behavior of a resource might trigger throttling in order to prevent overuse or service degradation. Throttling can occur at the application, tenant, or entire service level. Most commonly it occurs when an application has a large number of requests within or across tenants. Similarly, there are Azure AD service limits and restrictions that might affect the runtime behavior of applications.

Common reasons for resource isolation

Sometimes a group of resources should be isolated from other resources for security or other reasons, such as the resources have unique access requirements. You must determine which users and security principals should have resource access and in what roles. Reasons to isolate resources might include:

- Developer teams need the flexibility to safely iterate during the software development lifecycle of apps. But the development and testing of apps that write to Azure AD can potentially affect the Azure AD tenant through write operations. Some examples of this include:
    - New applications that may change Office 365 content such as SharePoint sites, OneDrive, MS Teams, etc.
    - Custom applications that can change data of users with MS Graph or similar APIs at scale (e.g., applications that are granted Directory.ReadWrite.All)
    - DevOps scripts that update large sets of objects as part of a deployment lifecycle.
    - Developers of Azure AD integrated apps need the ability to create user objects for testing, and those user objects should not have access to production resources.
- Non-production Azure resources and applications that may affect other resources. For example, a new beta version of a SaaS application may need to be isolated from the production instance of the application and production user objects
- Secret resources that should be shielded from discovery, enumeration, or takeover from existing administrators for regulatory or business critical reasons.

Configuration in a tenant

Configuration settings in Azure AD can impact any resource in the Azure AD tenant through targeted, or tenant-wide management actions. Examples of tenant-wide settings include:

- External identities: Global administrators for the tenant identify and control the external identities that can be provisioned in the tenant.
    - Whether to allow external identities in the tenant.
    - From which domain(s) external identities can be added.
    - Whether users can invite users from other tenants.
- Named Locations: Global administrators can create named locations, which can then be used to
    - Block sign-ins from specific locations.
    - Trigger conditional access policies such as MFA.
- Allowed authentication methods: Global administrators set the authentication methods allowed for the tenant.
- Self-service options. Global Administrators set self-service options such as self-service-password reset and create Office 365 groups at the tenant level.

The implementation of some tenant-wide configurations can be scoped as long as they do not get overridden by global administration policies. For example:

- If the tenant is configured to allow external identities, a resource administrator can still exclude those identities from accessing a resource.
- If the tenant is configured to allow personal device registration, a resource administrator can exclude those devices from accessing specific resources.
- If named locations are configured, a resource administrator can configure policies either allowing or excluding access from those locations.

Common reasons for configuration isolation

Configurations, usually controlled by Global Administrators, affect resources. While some tenant-wide configuration can be scoped with policies to not apply or partially apply to a specific resource, others cannot. If a resource group has configuration needs that are unique, it will likely need to be isolated in a separate tenant.

Examples of configuration isolation scenarios include:

- Resources having requirements that conflict with existing tenant-wide security or collaboration postures. (for example allowed authentication types, device management policies, ability to self-service, identity proofing for external identities, etc.).
- Compliance requirements that scope certification to the entire environment, including all resources and the Azure AD tenant itself, especially when those requirements conflict with or must exclude other organizational resources.
- External user access requirements that conflict with production or sensitive resource policies.

Administration in a tenant

Identities with privileged roles in the Azure AD tenant have the visibility and permissions to execute the configuration tasks described in the previous sections. Administration includes both the administration of identity objects such as users, groups, and devices, and the scoped implementation of tenant-wide configurations for authentication, authorization, etc.

Administration of directory objects

Administrators manage how identity objects can access resources, and under what circumstances. They also can disable, delete, or modify directory objects based on their privileges.

Identity objects include:

- Organizational identities, such as the following, are represented by user objects:
    - Administrators
    - Organizational users
    - Organizational developers
    - Test users

- External identities represent users from outside the organization such as:
    - Partners, suppliers, or vendors that are provisioned with accounts local to the organization environment
    - Partners, suppliers, or vendors that are provisioned via Azure B2B collaboration
- Groups are represented by objects such as:
    - Security groups
    - Office 365 groups
- Devices are represented by objects such as:
    - Hybrid Azure AD joined devices (On-premises computers synchronized from on-premises Active Directory)
    - Azure AD joined devices
    - Azure AD registered mobile devices used by employees to access their workplace applications.

In a hybrid environment, identities are typically synchronized from the on-premises Active Directory environment using Azure AD Connect.

Administration of identity services

Administrators with appropriate permissions can also manage how tenant-wide policies are implemented at the level of resource groups, security groups, or applications. When considering administration of resources, keep the following in mind. Each can be a reason to keep resources together, or to isolate them.

- An identity assigned an Authentication Administrator role can require non-administrators to reregister for MFA or FIDO authentication.
- A Conditional Access (CA) Administrator can create CA policies that require users signing-in to specific apps to do so only from organization-owned devices. They can also scope configurations. For example, even if external identities are allowed in the tenant, they can exclude those identities from accessing a resource.
- A Cloud Application Administrator can consent to application permissions on behalf of all users.
- A Global Administrator can take control of a subscription.

Common reasons for administrative isolation

Who should have the ability to administer the environment and its resources? There are times when administrators of one environment must not have access to another environment. Examples include:

- Separation of tenant-wide administrative responsibilities to further mitigate the risk of security and operational errors affecting critical resources.
- Regulations that constrain who can administer the environment based on conditions such as country of citizenship, country of residency, clearance level, etc. that cannot be accommodated with existing staff.

Security and operational considerations

Given the interdependence between an Azure AD tenant and its resources, it is critical to understand the security and operational risks of compromise or error.

Identity compromise. Within the boundary of a tenant, any identity can be assigned any role, given the one providing access has sufficient privileges. While the impact of compromised non-privileged identities is largely contained, compromised administrators can have broad impact. To mitigate risk of identity compromise, or bad actors, implement tiered administration and ensure that you follow principles of least privilege for Azure AD Administrator Roles. Similarly, ensure that you create CA policies that specifically excluded test accounts and test service principles from accessing resources outside of the test applications.

Trusting resource compromise. Human identities are not the only security consideration. Any compromised component of the Azure AD tenant can impact trusting resources based on its level of permissions at the tenant and resource level. The impact of a compromised component of an Azure AD trusting resource is determined by the privileges of the resource; resources that are deeply integrated with the directory to perform write operations can have profound impact in the entire tenant.

Application development. Early stages of the development lifecycle for applications with writing privileges to Azure AD, where bugs can unintentionally write changes to the Azure AD objects, present a risk. Follow Microsoft Identity platform best practices during development to mitigate these risks.

Operational error. A security incident can occur not only due to bad actors, but also because of an operational error by tenant administrators or the resource owners. These risks occur in any architecture. Mitigate these risks with tiered administration, following principles of least privilege, and following best practices before trying to mitigate by using a separate tenant.

In the following sections, we present three isolation architectures that can address these requirements. They are presented from least to most isolated.

Delegating administration in a single tenant

Many separation scenarios can be achieved within a single tenant. If possible, we recommend that you delegate administration to separate environments within a single tenant to provide the best productivity and collaboration experience for your organization.

Outcomes

Resource separation

With Azure AD directory roles, security groups, conditional access policies, Azure resource groups, Azure management groups, and other controls, you can restrict resource access to specific users, groups, and service principals. Resources can be managed by separate administrators, and have separate users, permissions, and access requirements.

If a set of resources require unique tenant-wide settings, or there is minimal risk tolerance for unauthorized access by tenant members, or critical impact could be caused by configuration changes, you must achieve isolation in multiple tenants.

Configuration separation

In some cases, resources such as applications have dependencies on tenant-wide configurations like authentication methods, named locations, and user risk profiles. You should consider these dependencies when isolating resources. Global administrators can configure the resource settings and tenant-wide settings that affect resources.

If a set of resources require unique tenant-wide settings, or the tenant's settings must be administered by a different entity, you must achieve isolation with multiple tenants.

Administrative separation

With Azure AD delegated administration, you can segregate the administration of resources such as applications and APIs, users and groups, resource groups, and conditional access policies.

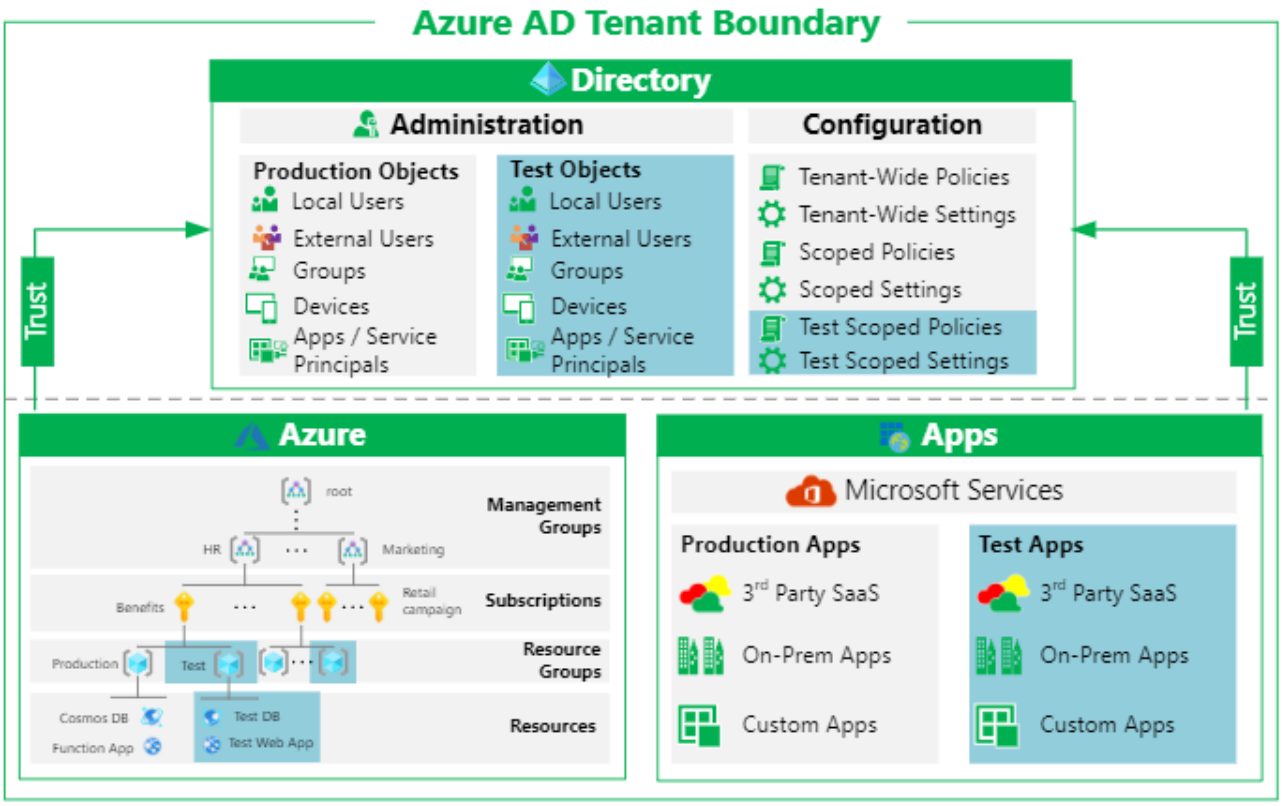
Global administrators can discover and obtain full access to any trusting resources. You can set up auditing and alerts to know when an administrator changes a resource if they are authenticated.

If you must ensure that global administrators are unable to manage a specific resource, you must isolate that resource in a separate tenant with separate global administrators.

Common usage

One of the most common uses for multiple environments in a single tenant is to segregate production from non-production resources. Within a single tenant, development teams and application owners can create and manage a separate environment with test apps, test users and groups, and test policies for those objects; similarly, they can create non-production instances of Azure resources and trusted apps.

The following diagram illustrates the non-production environments and the production environment.



Azure AD Tenant Boundary

In this diagram, there are non-production Azure resources and non-production instances Azure AD integrated applications with equivalent non-production directory objects. In this example, the non-production resources in the directory are used for testing purposes.

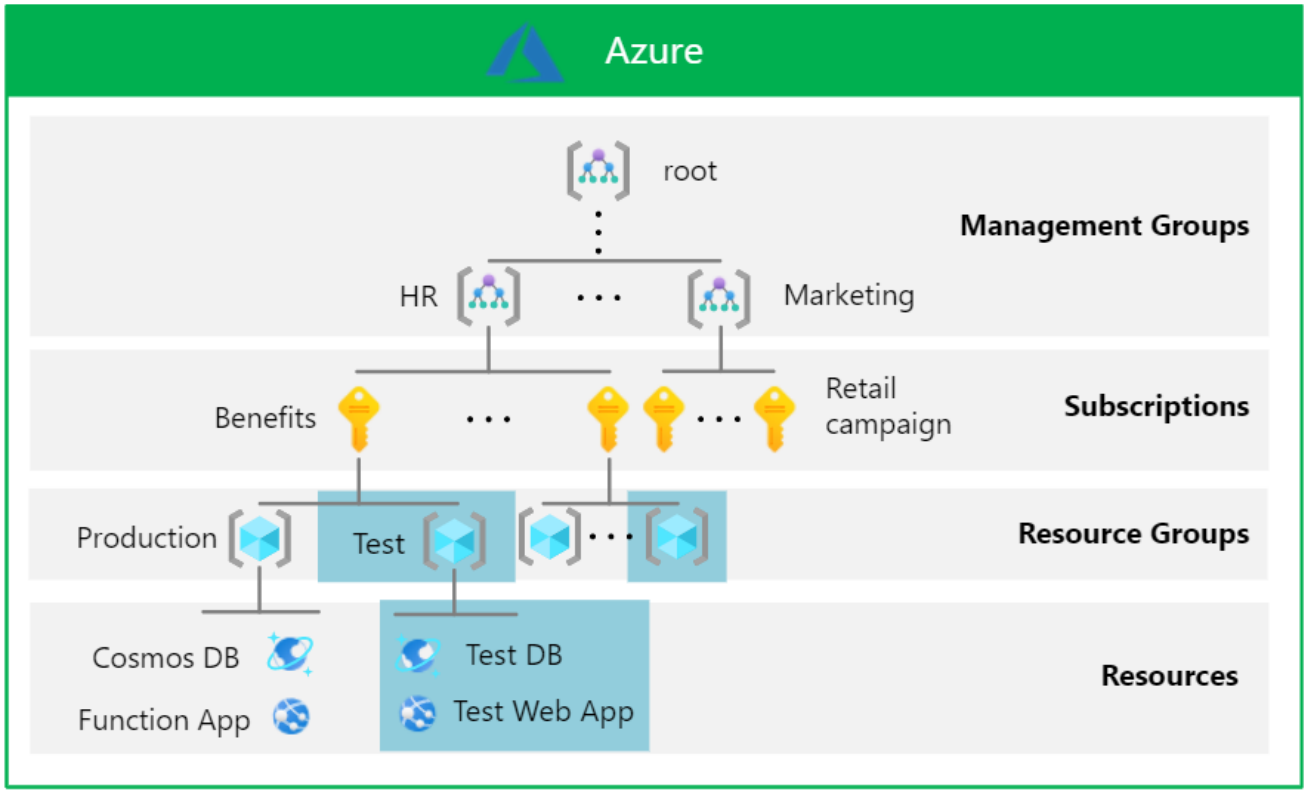
Azure RBAC role assignments allow scoped administration of Azure resources. Similarly, Azure AD allows granular management of Azure AD trusting applications through multiple capabilities such as conditional access, user and group filtering, and application assignment.

Delegated administration in a single tenant

Scoped management for Azure resources

Azure RBAC allows you to design an administration model with granular scopes and surface area. Consider the management hierarchy in the following example:

Note: There are multiple ways to define the management hierarchy based on an organization's individual requirements, constraints, and goals. For more information, consult the Cloud Adoption Framework guidance on how to Organize Azure Resources.



Resource isolation in a single tenant

- **Individual resources**. You can assign roles to specific resources so that they don't impact any other resources. In the example above, the Benefits engineering team can assign a data analyst the Cosmos DB Account Reader role just for the test instance of the Cosmos DB, without interfering with the test web app, or any production resource.
- **Resource group**. You can assign roles to specific resource groups so that they don't impact any other resource groups. In the example above, the Benefits engineering team can assign the Contributor role to the test lead so they can manage the test DB and the test web app, or to add more resources.
- **Subscription**. You can assign roles to a specific subscription to prevent it from impacting any other resource groups. In the example above, the HR team can assign the Reader role for the Benefits subscription, without reading any other HR subscription, or a subscription from any other team.
- **Management group**. You can assign roles to specific management groups so that they don't impact any other management groups. In the scenario above, the HR team can define an Azure Policy to audit the regions where resources are deployed across all HR subscriptions.

  For more information, see Azure built-in roles and What is Azure role-based access control (Azure RBAC)?.

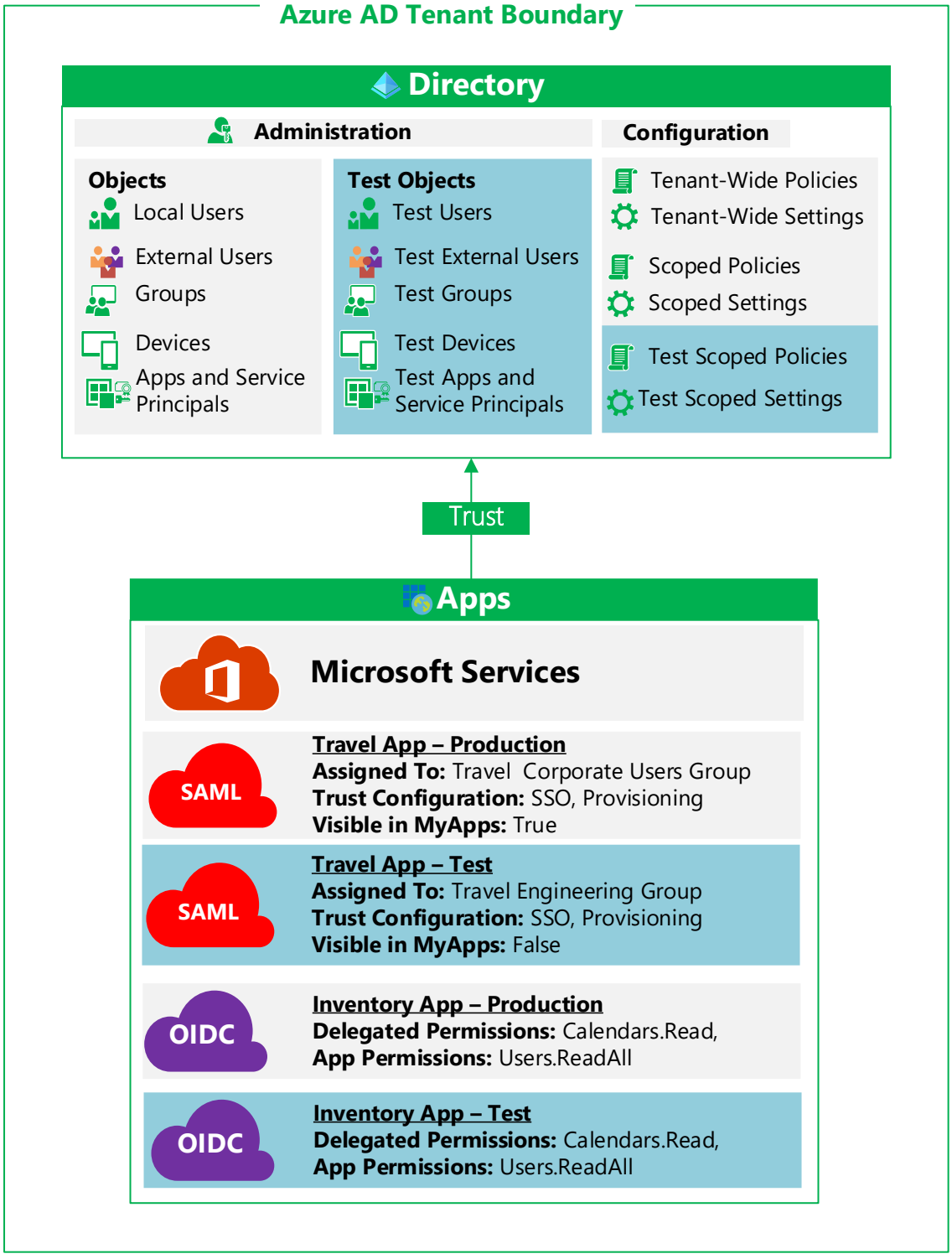
This is a hierarchical structure, so the higher up in the hierarchy, the more scope, visibility, and impact there is to lower levels. Top-level scopes affect all Azure resources in the Azure AD tenant boundary.

- The root management group defines Azure Policies and RBAC role assignments that will be applied to all subscriptions and resources.
- Global Administrators can elevate access to all subscriptions and management groups.

It is important to plan for other dimensions of resource isolation such as networking. For general guidance on Azure networking, see Azure best practices for network security. Infrastructure as a Service (IaaS) workloads have special scenarios where both identity and resource isolation need to be part of the overall design and strategy. See the appendix for Identity Considerations for IaaS Solutions in Azure for more details.

Scoped management for Azure AD trusting applications

The pattern to scope management of Azure AD trusting applications is outlined in the diagram below.



Azure AD supports configuring multiple instances of custom and SaaS apps, but not most Microsoft services, against the same directory with independent user assignments. The above example contains both a production and a test version of the travel app. You can deploy pre-production versions against the corporate tenant to achieve app-specific configuration and policy separation that enables workload owners to perform testing with

their corporate credentials. Non-production directory objects such as test users and test groups are associated to the non-production application with separate ownership of those objects.

There are tenant-wide aspects that affect all trusting applications in the Azure AD tenant boundary including:

- Global Administrators can manage all tenant-wide settings.
- Other directory roles such as User Administrator, Administrator, and Conditional Access Administrators can manage tenant-wide configuration within the scope of the role.
- Configuration settings such authentication methods allowed, hybrid configurations, B2B collaboration allow-listing of domains, and named locations are tenant wide.

IMPORTANT: The lifecycle of Microsoft SaaS services such as Office 365, Microsoft Dynamics, and Microsoft Exchange are bound to the Azure AD tenant. As a result, multiple instances of these services necessarily require multiple Azure AD tenants. Check the documentation for individual services to learn more about specific management scoping capabilities.

Resource isolation with multiple tenants

There are specific scenarios when delegating administration within a single tenant boundary will not meet your needs. In this section, we will discuss requirements that may drive you to create a multi-tenant architecture. **Multi-tenant architectures increase management overhead and complexity and should be used with caution. We recommend using a single tenant if your needs can be met with that architecture.**

A separate tenant creates a new boundary, and therefore decoupled management of Azure AD directory roles, directory objects, conditional access policies, Azure resource groups, Azure management groups, and other controls as described in previous sections.

A separate tenant is useful for an organization's IT department to validate tenant-wide changes in Microsoft services such as, Intune, Azure AD Connect, or a hybrid authentication configuration while protecting an organization's users and resources. This includes testing service configurations that might have tenant-wide impact and cannot be scoped to a subset of users in the production tenant.

Deploying a non-production environment in a separate tenant during early phases of a development cycle may be necessary during development of custom applications that can change data of production user objects with MS Graph or similar APIs (e.g., applications that are granted Directory.ReadWrite.All, or similar wide scope).

Outcomes

In addition to the outcomes achieved with a single tenant architecture as described above, organizations can fully decouple the resource and tenant interactions as described below:

Resource Separation

- **Visibility**. Resources in a separate tenant cannot be discovered or enumerated by users and administrators in other tenants. Similarly, usage reports and audit logs are contained within the new tenant boundary. This separation of visibility allows organizations to manage resources needed for confidential projects.
- **Object Footprint.** Applications that write to Azure AD and/or other Microsoft Online services through Microsoft Graph or other management interfaces can operate in a separate object space. This enables

development teams to perform tests during the software development lifecycle without affecting other tenants.

- Quotas. Consumption of tenant-wide Azure Quotas and Limits is separated from that of the other tenants.

Configuration separation

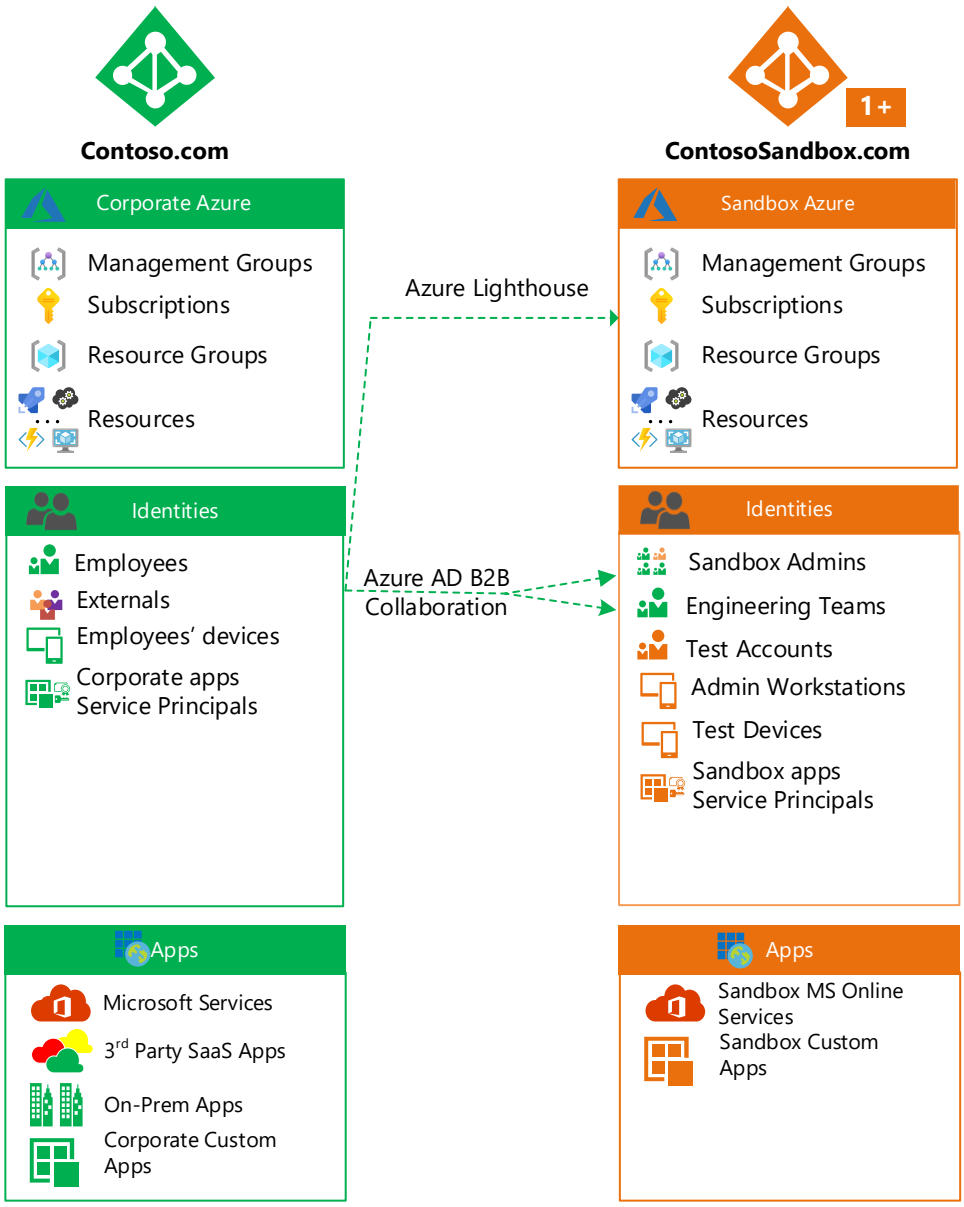
A new tenant provides a separate set of tenant-wide settings that can accommodate resources and trusting applications that have requirements that need different configurations at the tenant level. Additionally, a new tenant provides a new set of Microsoft Online services such as Office 365.

Administrative separation

A new tenant boundary involves a separate set of Azure AD directory roles, which enables you to configure different sets of administrators.

Common usage

The following diagram illustrates a common usage for resource isolation in multiple tenants: a pre-production or “sandbox” environment that requires more separation than can be achieved with delegated administration in a single tenant.



Contoso is an organization that augmented their corporate tenant architecture with a pre-production tenant called ContosoSandbox.com. The sandbox tenant is used to support ongoing development of enterprise solutions that write to Azure AD and Office 365 using Microsoft Graph. These solutions will subsequently be deployed in the corporate tenant.

The sandbox tenant is brought online to prevent those applications under development from impacting production systems either directly or indirectly or by consuming tenant resources and affecting quotas or throttling.

Developers require access to the sandbox tenant during the development lifecycle, ideally with self-service access requiring additional permissions that are restricted in the production environment. Examples of these additional permission might include creating, deleting, and updating user accounts, registering applications, provisioning and deprovisioning Azure resources, and changes to policies or overall configuration of the environment.

In this example, Contoso used Azure AD B2B Collaboration to provision users from the corporate tenant to enable users to manage and access resources in applications in the sandbox tenant without managing multiple credentials. While this capability is mostly oriented to cross-organization collaboration scenarios, enterprises with multiple tenants like Contoso can leverage this capability to avoid additional credential lifecycle administration and user experience complexities.

Contoso enabled Azure Lighthouse to manage Azure resources in the sandbox tenant.

Multi-tenant resource isolation

A new tenant provides the ability to have a separate set of administrators. Organizations can choose to use corporate identities through Azure AD B2B collaboration. Similarly, organizations can implement Azure Lighthouse for cross-tenant management of Azure resources so that non-production Azure subscriptions can be managed by identities in the production counterpart. Azure Lighthouse cannot be used to manage services outside of Azure, such as Intune or Microsoft Endpoint Manager.

This will allow users to continue to use their corporate credentials, while achieving the separation outcomes described above.

Azure AD B2B collaboration in sandbox tenants should be configured to allow only identities from the corporate environment to be onboarded using Azure B2B allow/deny lists.

IMPORTANT*:* Multi-tenant architectures with external identity access enabled provide only resource isolation, but do not enable identity isolation. *Resource isolation using Azure AD B2B collaboration and Azure Lighthouse do not mitigate risks related to identities*. Because the sandbox environment shares identities with the corporate environment, the following scenarios are applicable to the sandbox tenant:

- A malicious actor that compromises a user, a device, or hybrid infrastructure in the corporate tenant, and is invited into the sandbox tenant, might gain access to the sandbox tenant's apps and resources.
- An operational error (e.g. user account deletion or credential revocation) in the corporate tenant might impact the access of an invited user into the sandbox tenant.

You must do the risk analysis and potentially consider identity isolation through multiple tenants for business critical resources that require a highly defensive approach.

Directory Objects

Your tenant used to isolate resources may contain the same types of objects, azure resources, and trusting applications as your primary tenant.

You may need to provision the following object types:

Users and groups: Identities needed by solution engineering teams, such as:

- Sandbox environment administrators.
- Technical owners of applications.
- Line-of-business application developers.
- Test end-user accounts.

These identities might be provisioned for:

- Employees who come with their corporate account through Azure AD B2B collaboration.
- Employees who need local accounts for administration, emergency administrative access, or other technical reasons.

Customers who have or require non-production Active Directory on-premises can also synchronize their on-premises identities to the sandbox tenant if needed by the underlying resources and applications.

Devices: This tenant contains a reduced number of devices to the extent that are needed in the solution engineering cycle:

- Administration workstations
- Non-production computers and mobile devices needed for development, testing, and documentation

Applications

Azure AD integrated applications: Application objects and service principals for:

- Test instances of the applications that are deployed in production (e.g. applications that write to Azure AD and Microsoft online services).
- Infrastructure services to manage and maintain the non-production tenant, potentially a subset of the solutions available in the corporate tenant.

Microsoft Online Services:

- Typically, the team that owns the Microsoft Online Services in production should be the one owning the non-production instance of those services.
- Administrators of non-production test environments should not be provisioning Microsoft Online Services unless those services are specifically being tested. This avoids inappropriate use of Microsoft services, for example setting up production Sharepoint sites in a test environment.
- Similarly, provisioning of Microsoft Online services that can be initiated by end users (also known as ad-hoc subscriptions) should be locked down. For more information, see What is self-service sign-up for Azure Active Directory?.

Azure resources

Any Azure resources needed by trusting applications may also be deployed. For example, databases, virtual machines, containers, Azure functions, etc.

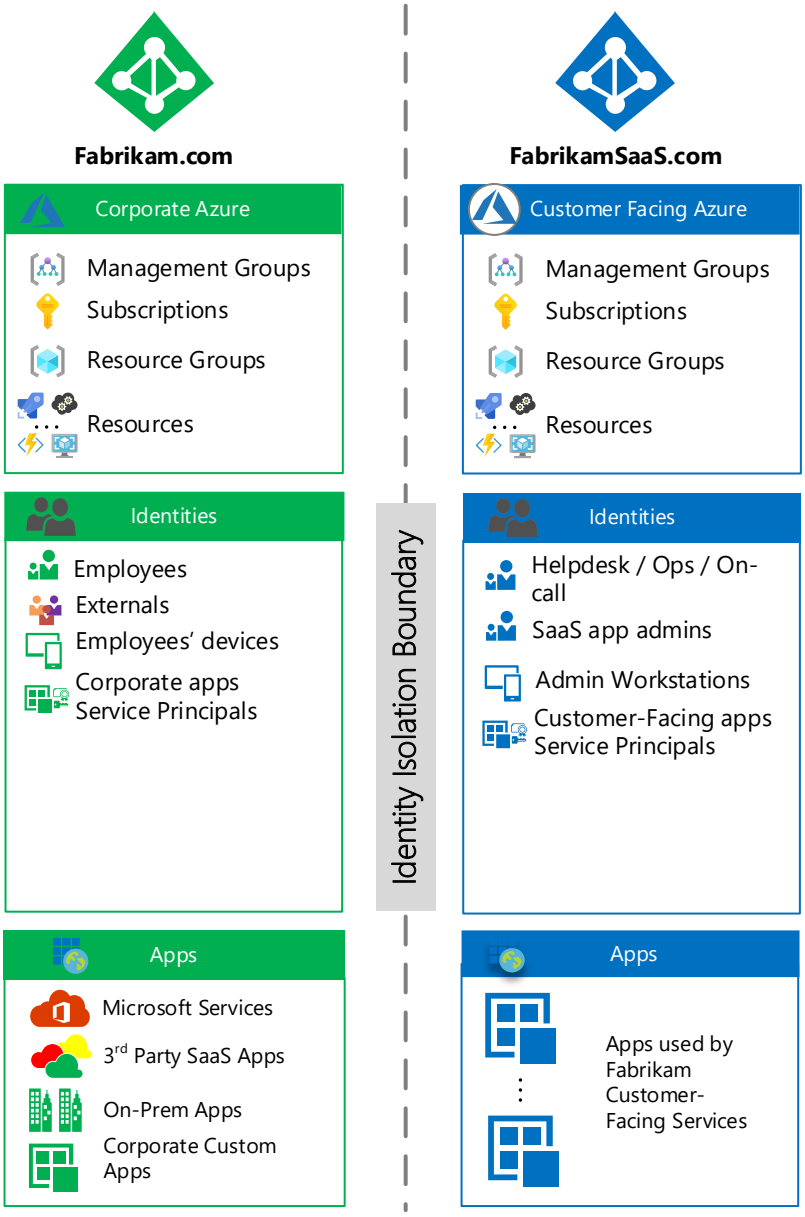
Resource and identity isolation with multiple tenants

Outcomes

There are limited situations where resource isolation alone will not meet your requirements. You can isolate both resources and identities in a multi-tenant architecture by *disabling all cross-tenant collaboration capabilities and effectively building a separate identity boundary*. This approach is a defense against operational errors and compromise of user identities, devices, or hybrid infrastructure in corporate tenants.

Common usage

A separate identity boundary is typically used for business-critical applications and resources such as customer-facing services. In this scenario, Fabrikam has decided to create a separate tenant for their customer-facing SaaS product to avoid the risk of employee identity compromise affecting their SaaS customers. The following diagram illustrates this architecture:



The FabrikamSaaS tenant contains the environments used for applications that are offered to customers as part of Fabrikam's business model.

Directory Objects

The directory objects in FabrikamSaas are as follows:

Users and groups: Identities needed by solution IT teams, customer support staff, or other necessary personnel are created within the SaaS tenant. To preserve isolation, only local accounts are used, and Azure AD B2B collaboration is not enabled.

Azure AD B2C directory objects: If the tenant environments are accessed by customers, it may contain an Azure AD B2C tenant and its associated identity objects. Subscriptions that hold these directories are good candidates for an isolated consumer-facing environment.

Devices: This tenant contains a reduced number of devices; only those that are needed to run customer-facing solutions:

- Secure administration workstations.
- Support personnel workstations (this can include engineers who are "on call" as described above).

Applications

Azure AD integrated applications: Application objects and service principals for:

- Production applications (e.g., multi-tenant application definitions).
- Infrastructure services to manage and maintain the customer-facing environment.

Azure Resources: Hosts the IaaS, PaaS and SaaS resources of the customer-facing production instances. For more information, see Identity considerations for Infrastructure as a Service in the appendix.

Best practices for all isolation architectures

The following are design considerations you should consider for all isolation configurations. Throughout this section you will find many links (they open in a new window). We link to content, rather than duplicate it here, so you will always have access to the most up-to-date information.

For general guidance on how to configure Azure AD tenants (isolated or not), please refer to the Azure Active Directory feature deployment guide.

Isolation Security Principles

When designing isolated environments, it is important to consider the following principles:

- **Use only modern authentication.** Applications deployed in isolated environments must use claims-based modern authentication (e.g., SAML, OAuth, OAuth2, and OpenID Connect) to leverage capabilities such as federation, Azure AD B2B collaboration, delegation, and the consent framework. This way, legacy applications that have dependency on legacy authentication methods such as NTLM will not carry forward in isolated environments.
- **Enforce strong authentication**. Strong authentication must always be used when accessing the isolated environment services and infrastructure. Whenever possible, passwordless authentication such as Windows for Business Hello or a FIDO2 security keys should be used.
- **Deploy secure workstations.** Secure workstations provide the mechanism to ensure that the platform and the identity that platform represents is properly attested and secured against exploitation.
- **Eliminate legacy trust mechanisms**. Isolated directories and services should not establish trust relationships with other environments through legacy mechanisms such as Active Directory trusts. All trusts between environments should be established with modern constructs such as federation and claims-based identity.
- **Isolate services**. Minimize the surface attack area by protecting underlying identities and service infrastructure from exposure. Enable access only through modern authentication for services and secure remote access (also protected by modern authentication) for the infrastructure.

In addition to the guidance in the Azure Active Directory general operations guide, we also recommend the following considerations for isolated environments.

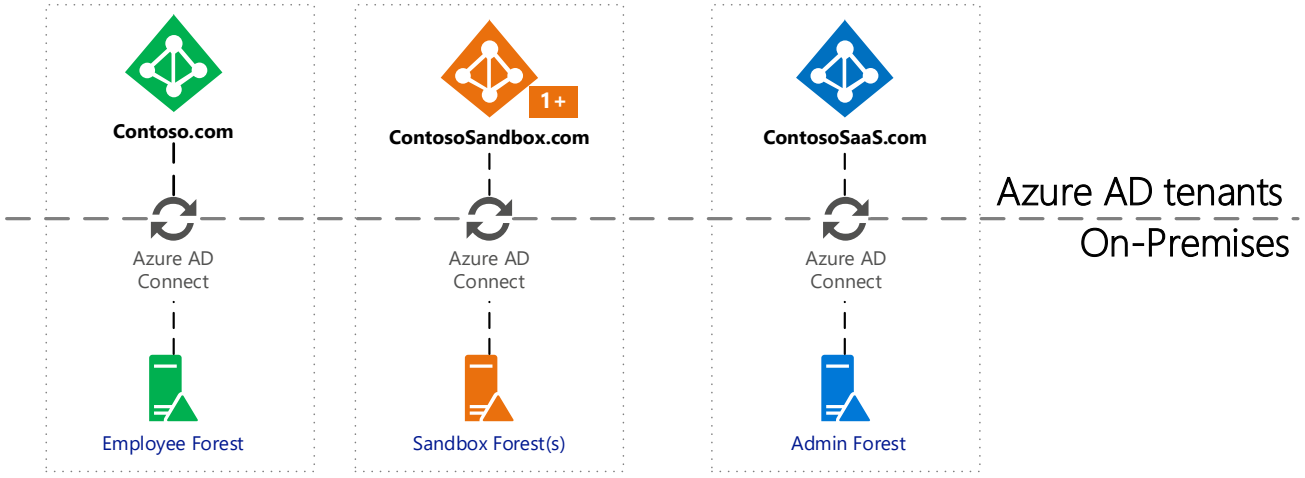
Human identity provisioning

Privileged Accounts

Provision accounts in the isolated environment for administrative personnel and IT teams who will be operating the environment. This will enable you to add stronger security policies such as device-based access control for secure workstations. As discussed in previous sections, non-production environments can potentially utilize Azure AD B2B collaboration to onboard privileged accounts to the non-production tenants using the same posture and security controls designed for privileged access in their production environment.

Cloud-only accounts are the simplest way to provision human identities in an Azure AD tenant and it is a good fit for greenfield environments. However, if there is an existing on-premises infrastructure that corresponds to the isolated environment (e.g., pre-production or management Active Directory forest), you should consider synchronizing identities from there, especially if the on-premises infrastructure described above is also used

for IaaS solutions that require server access to manage the solution data plane. Synchronizing from isolated on-premises environments might also be needed if there are specific regulatory compliance requirements such as smart-card only authentication.



Synchronizing admin identities from on-premises

Note on B2B collaboration usage

There are no technical controls to do identity proofing for Azure AD B2B accounts. External identities provisioned with Azure AD B2B are bootstrapped with a single factor. The mitigation is for the organization to have a process to proof the required identities prior to a B2B invitation being issued, and regular access reviews of external identities to manage the lifecycle. Consider enabling a Conditional Access policy to control the MFA registration.

Non-human identity provisioning

Emergency Access Accounts

Provision emergency access accounts for "break glass"' scenarios where normal administrative accounts can't be used in the event you are accidentally locked out of your Azure AD organization. For on-premises environments using federation systems such as Active Directory Federation Services (AD FS) for authentication, maintain alternate cloud-only credentials for your global administrators to ensure service delivery in the event of an on-premises infrastructure outage.

Azure Managed Identities

Leverage Azure managed identities for Azure resources that require a service identity. Check the list of services that support managed identities when designing your Azure solutions.

If managed identities are not supported or not possible, consider provisioning service principal objects.

Hybrid Service Accounts

Some hybrid solutions might require access to both on-premises and cloud resources. An example of a use case would be an Identity Governance solution that uses a service account on premises for access to AD DS and requires access to Azure AD.

On-premises service accounts typically do not have the ability to log in interactively, which means that in cloud scenarios they cannot fulfill strong credential requirements such as Multi-Factor Authentication. In this scenario, don't synchronize those accounts to the cloud; instead provision a managed identity or service principal to connect to the cloud portion of the hybrid solution.

If there are technical constraints that don't make this possible and the same account must be used for both on-premises and cloud, then implement compensating controls such as Conditional Access to lock down the hybrid account to come from a specific network location.

Resource Assignment

An enterprise solution might be comprised of multiple Azure resources and its access should be managed and governed as a logical unit of assignment—a resource group. In that scenario, Azure AD security groups can be created and associated with the proper permissions and role assignment across all solution resources, so that adding or removing users from those groups results in allowing or denying access to the entire solution.

We recommend you use security groups to grant access to Microsoft services that rely on licensing to provide access (e.g., Dynamics 365, Power BI).

Azure AD cloud native groups can be natively governed from the cloud when combined with Azure AD access reviews and Azure AD entitlement management. Organizations who already have on-premises group governance tools can continue to use those tools and rely on identity synchronization with Azure AD Connect to reflect group membership changes.

Azure AD also supports direct user assignment to third-party SaaS services (e.g., Salesforce, Service Now) for single sign-on and identity provisioning. Direct assignments to resources can be natively governed from the cloud when combined with Azure AD access reviews and Azure AD entitlement management. Direct assignment might be a good fit for end-user facing assignment.

Some scenarios might require granting access to on-premises resources through on-premises Active Directory security groups. For those cases, consider the synchronization cycle to Azure AD when designing processes SLA.

Authentication management

This section describes the checks to perform and actions to take for credential management and access policies based on your organization's security posture.

Credential Management

Strong credentials

All human identities (local accounts and external identities provisioned through B2B collaboration) in the isolated environment must be provisioned with strong authentication credentials such as Azure Multi-Factor Authentication. Environments with an underlying on-premises infrastructure with strong authentication such as smart card authentication can continue leveraging smart card authentication in the cloud.

Passwordless credentials

A passwordless solution is the best solution for ensuring the most convenient *and* secure method of authentication. Passwordless credentials such as FIDO security keys and Windows Hello for Business are recommended for human identities with privileged roles.

Password Protection

If the environment is synchronized from an on-premises Active Directory forest, you should deploy Azure AD password protection to eliminate weak passwords in your organization. Azure AD smart lockout should also be used in hybrid or cloud-only environments to lock out bad actors who are trying to guess your users' passwords or use brute-force methods to get in.

Self-service password management

Users needing to change or reset their passwords is one of the biggest sources of volume and cost of help desk calls. In addition to cost, changing the password as a tool to mitigate a user risk is a fundamental step in improving the security posture of your organization. At a minimum, deploy Self-Service Password Management for human and test accounts with passwords to deflect help desk calls.

External Identities Passwords

By using Azure AD B2B collaboration, an invitation and redemption process lets external users such as partners, developers, and subcontractors use their own credentials to access your company's resources. This mitigates the need to introduce more passwords into the isolated tenants. Note, some applications, infrastructure, or workflows might require a local credential. Evaluate this on a case by case basis.

Service Principals Credentials

For scenarios where service principals are needed, use certificate credentials for service principals and fall back to client secrets as an exception.

In both cases, Azure Key Vault can be used to in conjunction with Azure managed identities, so that the runtime environment (e.g., an Azure function) can retrieve the credential from the key vault.

Check this example to create service principals with self-signed certificate for authentication of service principals with certificate credentials.

Access Policies

Below are some specific recommendations for Azure solutions. For general guidance on Conditional Access policies for individual environments, check the CA Best practices and Azure AD Operations Guide:

- Define Conditional Access policies for the Microsoft Azure Management cloud app to enforce identity security posture when accessing ARM. This should include controls on MFA and device-based controls to enable access only through secure workstations (more on this in the Privileged Roles section under Identity Governance).
- All applications onboarded to isolated environments must have explicit Conditional Access policies applied as part of the onboarding process.
- Define Conditional Access policies for security information registration that reflects a secure root of trust process on-premises (e.g., for workstations in physical locations, identifiable by IP addresses, that employees must visit in person for verification).
- Consider managing Conditional Access policies at scale with automation using MS Graph CA API. For example, you can use the API to configure, manage, and monitor CA policies consistently across tenants.

Authentication Challenges

- External identities provisioned with Azure AD B2B need to re-provision Multi-Factor Authentication credentials in the resource tenant, which means that onboarding to the system is bootstrapped with a single factor. The mitigation is for the organization to have a process to proof the user and credential risk profile prior to a B2B invitation being issued, as well as define Conditional Access to the registration process as discussed above.
- External identities provisioned with Azure AD B2B cannot honor device-based Conditional Access (e.g., the "compliant device" status in the corporate tenant does not carry forward to the same control in the isolated tenant), so the only effective control is Multi-Factor Authentication.
- Azure AD Conditional Access does not have affinity to specific devices to access applications such as ARM. To mitigate this, lock down the access to the management networks to only secure workstations, and then define Conditional Access policies to only allow access for privileged identities from compliant devices that originate from managed networks.
- Billing management applications such as Azure EA portal or MCA billing accounts are not represented as cloud applications for Conditional Access targeting. As a compensating control, define separate administration accounts and target Conditional Access policies to those accounts using an "All Apps" condition.

Identity Governance

Privileged Roles

Below are some identity governance principles to consider across all the tenant configurations for isolation.

- **No standing access:** No human identities should have standing access to perform privileged operations in isolated environments. Azure RBAC integrates with Azure AD Privileged Identity Management (PIM). PIM provides just-in-time activation determined by security gates such as Multi-Factor Authentication, approval workflow, and limited duration.
- **Number of admins:** Organizations should define minimum and maximum number of humans holding a privileged role to mitigate business continuity risks. With too few privileged roles, there may not be enough time-zone coverage. Mitigate security risks by having as few administrators as possible, following the least-privilege principle.
- **Least privileged access:** Identities should only be granted the permissions needed to perform the privileged operations per their role in the organization.
    - Azure RBAC custom roles allow designing least privileged roles based on organizational needs. We recommend that custom roles definitions are authored or reviewed by specialized security teams and mitigate risks of unintended excessive privileges. Authoring of custom roles can be audited through Azure Policy.
    - To mitigate accidental use of roles that are not meant for wider use in the organization, use Azure Policy to define explicitly which role definitions can be used to assign access. Learn more from this Github Sample.

- **Privileged access from secure workstations:** All privileged access should occur from secure, locked down devices. Separating these sensitive tasks and accounts from daily use workstations and devices protect privileged accounts from phishing attacks, application and OS vulnerabilities, various impersonation attacks, and credential theft attacks such as keystroke logging, Pass-the-Hash, and Pass-The-Ticket.

  Depending on your requirements, you can decide to build:

  - Azure-managed workstations using cloud endpoint management services
  - Privileged Access Workstations using on-premises endpoint management tools

- **Privileged role process guardrails:** Organizations must define processes and technical guardrails to ensure that privileged operations can be executed whenever needed while complying with regulatory requirements. Examples of guardrails criteria include:

  - Qualification of humans with privileged roles (e.g., full-time employee/vendor, clearance level, citizenship)
  - Explicit incompatibility of roles (also known as separation of duties). Examples include teams with Azure AD directory roles should not be responsible for managing Azure ARM privileged roles, etc.
  - Whether direct user or groups assignment are preferred for which roles.

Resource Access

- **Attestation:** Identities that hold privileged roles should be reviewed periodically to keep membership current and justified. Azure AD Access Reviews integrate with Azure RBAC roles, group memberships and Azure AD B2B external identities.
- **Lifecycle:** Privileged operations might require access to multiple resources such as line of business applications, SaaS Applications, and Azure resource groups and subscriptions. Azure AD Entitlement Management allows defining access packages that represent a set resources that is assigned to users as a unit, establish a validity period, approval workflows, etc.

Governance Challenges

- The Azure Enterprise (Azure EA) Agreement portal does not integrate with Azure RBAC or Conditional Access. The mitigation for this is to use dedicated administration accounts that can be targeted with policies and additional monitoring.
- The Azure EA Enterprise portal does not provide an audit log. To mitigate this, consider an automated governed process to provision subscriptions with the considerations described above and use dedicated EA accounts and audit the authentication logs.
- Microsoft Customer Agreement (MCA) roles do not integrate natively with PIM. To mitigate this, use dedicated MCA accounts and monitor usage of these accounts.
- Monitoring IAM assignments outside Azure AD PIM is not automated through Azure Policies. The mitigation is to not grant Subscription Owner or User Access Administrator roles to engineering teams. Instead create groups assigned to least privileged roles such as Contributor and delegate the management of those groups to engineering teams.

- Privileged roles in Azure AD B2C tenants are not integrated with Azure AD PIM. The mitigation is to create dedicated accounts in the organization's Azure AD tenant, onboard them in the Azure AD B2C tenant and apply conditional access policies to these dedicated administration accounts.
- Azure AD B2C tenant privileged roles are not integrated with Azure AD Access Reviews. The mitigation is to create dedicated accounts in the organization's Azure AD tenant, add these accounts to a group and perform regular access reviews on this group.
- There are no technical controls to subordinate the creation of tenants to an organization. The onboarding to the billing plane is a compensating control at the gate. This needs to be complemented with monitoring and alerts instead.
- There is no out-of-the box product to implement the subscription provisioning workflow recommended above. Organizations need to implement their own workflow.
- Azure Lighthouse is not integrated with Azure AD PIM.

Tenant and Subscription lifecycle management

Tenant Lifecycle

- We recommend implementing a process to request a new corporate Azure AD tenant. The process should account for:
    - Business justification to create it. Creating a new Azure AD tenant will increase complexity significantly, so it is key to ascertain if a new tenant is necessary.
    - The Azure cloud in which it should be created (e.g., Commercial, Government, etc.).
    - Whether this is production or not production
    - Directory data residency requirements
    - Global Administrators who will manage it
    - Training and understanding of common security requirements.
- Upon approval, the Azure AD tenant will be created, configured with necessary baseline controls, and onboarded in the billing plane, monitoring, etc.
- Regular review of the Azure AD tenants in the billing plane needs to be implemented to detect and discover tenant creation outside the governed process. Refer to the Inventory and Visibility section of this document for further details.
- Azure AD B2C tenant creation can be controlled using Azure Policy. The policy executes when an Azure subscription is associated to the B2C tenant (a pre-requisite for billing). Customers can limit the creation of Azure AD B2C tenants to specific management groups.

Subscription Lifecycle

Below are some considerations when designing a governed subscription lifecycle process:

- Define a taxonomy of applications and solutions that require Azure resources. All teams requesting subscriptions should supply their "product identifier" when requesting subscriptions. This information taxonomy will determine:
    - Azure AD tenant to provision the subscription
    - Azure EA account to use for subscription creation
    - Naming convention
    - Management group assignment
    - Other aspects such as tagging, cross-charging, product-view usage, etc.
- Do not allow ad-hoc subscription creation through the portals or by other means. Instead consider managing subscriptions programmatically using Azure ARM and pulling consumption and billing reports programmatically. This can help limit subscription provisioning to authorized users and enforce your policy and taxonomy goals.
- When a subscription is provisioned, create Azure AD cloud groups to hold standard ARM Roles needed by application teams such as Contributor, Reader and approved custom roles. This enables you to manage Azure RBAC role assignments with governed privileged access at scale.
    1. Configure the groups to become eligible for Azure RBAC roles using Azure AD PIM with the corresponding controls such as activation policy, access reviews, approvers, etc.
    2. Then delegate the management of the groups to solution owners.
    3. As a guardrail, do not assign product owners to User Access Administrator or Owner roles to avoid inadvertent direct assignment of roles outside Azure AD PIM, or potentially changing the subscription to a different tenant altogether.
    4. For customers who choose to enable cross-tenant subscription management in non-production tenants through Azure Lighthouse, make sure that the same access policies from the production privileged account (e.g., privileged access only from secured workstations) are enforced when authenticating to manage subscriptions.
- If your organization has pre-approved reference architectures, the subscription provisioning can be integrated with resource deployment tools such as Azure Blueprints or Terraform.
- Given the tenant affinity to Azure Subscriptions, subscription provisioning should be aware of multiple identities for the same human actor (employee, partner, vendor, etc.) across multiple tenants and assign access accordingly.

Azure AD B2C Tenants

- In an Azure AD B2C tenant, the built-in roles do not support PIM. To increase security, we recommend using Azure AD B2B collaboration to onboard the engineering teams managing Customer Identity Access Management (CIAM) from your Azure tenant, and assign them to Azure AD B2C privileged roles.
- Following the emergency access guidelines for Azure AD above, consider creating equivalent emergency access accounts in addition to the external administrators described above.

- We recommend the logical ownership of the underlying Azure AD subscription of the B2C tenant aligns with the CIAM engineering teams, in the same way that the rest of Azure subscriptions are used for the B2C solutions.

Operations

Below are additional operational considerations for Azure AD, specific to multiple isolated environments. Check the Azure Cloud Adoption Framework, Azure Security Benchmark and Azure AD Operations guide for detailed guidance to operate individual environments.

Cross-Environment Roles and Responsibilities

Enterprise-wide SecOps architecture. Members of operations and security teams from all environments in the organization should jointly define the following:

- Principles to define when environments need to be created, consolidated, or deprecated.
- Principles to define management group hierarchy on each environment.
- Billing plane (EA portal / MCA) security posture, operational posture, and delegation approach.
- Tenant creation process.
- Enterprise application taxonomy.
- Azure subscription provisioning process.
- Isolation and administration autonomy boundaries and risk assessment across teams and environments.
- Common baseline configuration and security controls (technical and compensating) and operational baselines to be used in all environments.
- Common standard operational procedures and tooling that spans multiple environments (e.g., monitoring, provisioning).
- Agreed upon delegation of roles across multiple environments.
- Segregation of duty across environments.
- Common supply chain management for privileged workstations.
- Naming conventions.
- Cross-environment correlation mechanisms.

Tenant creation. A specific team should own creating the tenant following standardized procedures defined by enterprise-wide SecOps architecture. This includes:

- Underlying license provisioning (e.g., Microsoft 365).
- Onboarding to corporate billing plan (e.g., Azure EA or MCA).
- Creation of Azure management group hierarchy.

- Configuration of management policies for various perimeters including identity, data protection, Azure, etc.
- Deployment of security stack per agreed upon cybersecurity architecture, including diagnostic settings, SIEM onboarding, CASB onboarding, PIM onboarding, etc.
- Configuration of Azure AD roles based on agreed upon delegation.
- Configuration and distribution of initial privileged workstations.
- Provisioning emergency access accounts.
- Configuration of identity provisioning stack.

Cross-environment tooling architecture. Some tools such as identity provisioning and source control pipelines might need to work across multiple environments. These tools should be considered critical to the infrastructure and must be architected, designed, implemented, and managed as such. As a result, architects from all environments should be involved whenever cross-environment tools need to be defined.

Inventory and Visibility

Azure Subscription Discovery. For each discovered tenant, an Azure AD global administrator can elevate access to gain visibility of all subscriptions in the environment. This elevation will assign the global administrator the User Access Administrator built-in role at the root management group.

Enabling Read Access to discover resources. Management groups enable RBAC assignment at scale across multiple subscriptions. Customers can grant a Reader role to a centralized IT team by configuring a role assignment in the root management group which will propagate to all subscriptions in the environment.

Resource Discovery. After gaining resource Read access in the environment, Azure Resource Graph can be used to query resources in the environment.

Logging and Monitoring

Central Security Log Management. Ingest logs from each environment in a centralized way, following consistent best practices across environments (e.g., diagnostics settings, log retention, SIEM ingestion, etc.). Azure Monitor can be used to ingest logs from different sources such as endpoint devices, network, operating systems' security logs, etc.

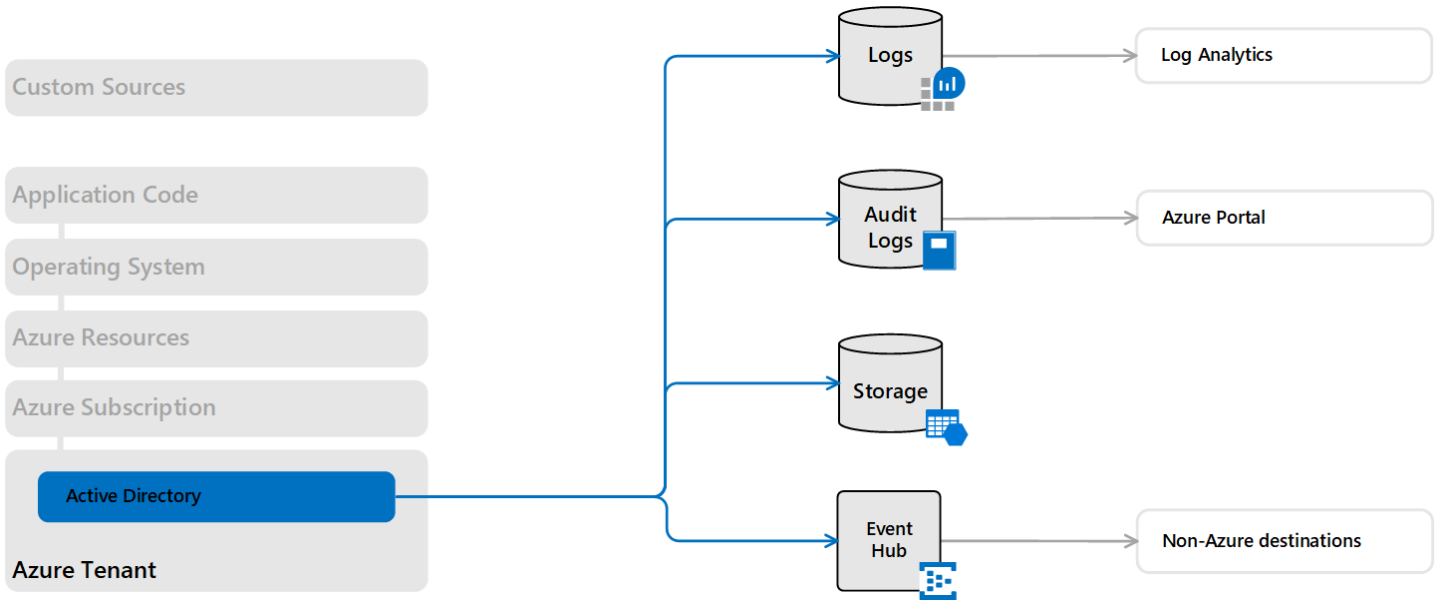
Some environments might have regulatory requirements that limit which data (if any) can leave a given environment. If centralized monitoring across environments is not possible, teams should have operational procedures to correlate activities of identities across environments for auditing and forensics purposes such as cross-environment lateral movement attempts. It is recommended that the object unique identifiers human identities belonging to the same person is discoverable, potentially as part of the identity provisioning systems.

The log strategy must include the following Azure AD logs for each tenant used in the organization:

- Sign-in activity
- Audit logs
- Risk events

Azure AD provides Azure Monitor integration for the sign-in activity log and audit logs. Risk events can be ingested through Microsoft Graph API.

The diagram below shows the different data sources that need to be incorporated as part of the monitoring strategy:



Monitoring strategy

Azure AD B2C tenants can be integrated with Azure Monitor. We recommend monitoring of Azure AD B2C using the same criteria discussed above for Azure AD.

Subscriptions that have enabled cross-tenant management with Azure Lighthouse can enable cross-tenant monitoring if the logs are collected by Azure Monitor. The corresponding Log Analytics workspaces can reside in the resource tenant and can be analyzed centrally in the managing tenant using Azure Monitor workbooks. To learn more, check Monitor delegated resources at scale - Azure Lighthouse.

Hybrid Infrastructure OS Security Logs

All hybrid identity infrastructure OS logs should be archived and carefully monitored as a Tier 0 system, given the surface area implications. This includes:

- AD FS servers and Web Application Proxy
- Azure AD Connect
- Application Proxy Agents
- Password write-back agents
- Password Protection Gateway machines
- NPS that have the Azure MFA RADIUS extension

Azure AD Connect Health must be deployed to monitor identity synchronization and federation (when applicable) for all environments.

Log Storage Retention. All environments should have a cohesive log storage retention strategy, design, and implementation to facilitate a consistent toolset (e.g., SIEM systems such as Azure Sentinel), common queries, investigation, and forensics playbooks. Azure Policy can be used to set up diagnostic settings.

Monitoring and Log Reviewing. The operational tasks around identity monitoring should be consistent and have owners in each environment. As described above, strive to consolidate these responsibilities across environments to the extent allowed by regulatory compliance and isolation requirements.

The following scenarios must be explicitly monitored and investigated:

- **Suspicious activity**: All Azure AD risk events should be monitored for suspicious activity. All tenants should define the network named locations to avoid noisy detections on location-based signals. Azure AD Identity Protection is natively integrated with Azure Security Center. It is recommended that any risk detection investigation includes all the environments the identity is provisioned (e.g., if a human identity has an active risk detection in the corporate tenant, the team operating the customer facing tenant should also investigate the activity of the corresponding account in that environment).
- **User Entity Behavioral Analytics (UEBA) alerts**: UEBA should be used to get insightful information based on anomaly detection. Microsoft Cloud App Discovery (MCAS) provides UEBA in the cloud. Customers can integrate on-prem UEBA from Azure ATP. MCAS reads signals from Azure AD Identity Protection.
- **Emergency access accounts activity**: Any access using emergency access accounts should be monitored and alerts created for investigations. This monitoring must include:
    - Sign-ins.
    - Credential management.
    - Any updates on group memberships.
    - Application Assignments.
- **Billing management accounts**: Given the sensitivity of accounts with billing management roles in Azure EA or MCA, and their significant privilege, it is strongly recommended to monitor and alert:
    - Sign in attempts by accounts with billing roles.
    - Any attempt to authenticate to applications other than the EA Portal.
    - Any attempt to authenticate to applications other than Azure Resource Management if using dedicated accounts for MCA billing tasks.
    - Assignment to Azure resources using dedicated accounts for MCA billing tasks.
- **Privileged role activity**: Configure and review security alerts generated by Azure AD PIM. If locking down direct RBAC assignments is not fully enforceable with technical controls (e.g., Owner role has to be granted to product teams to do their job), then monitor direct assignment of privileged roles outside PIM by generating alerts whenever a user is assigned directly to access the subscription with Azure RBAC.
- **Classic role assignments:** Organizations should use the modern Azure RBAC role infrastructure instead of the classic roles. As a result, the following events should be monitored:

    - Assignment to classic roles at the subscription level
- Tenant-wide configurations: Any tenant-wide configuration service should generate alerts in the system.
    - Updating Custom Domains
    - Azure AD B2B allow/block list
    - Azure AD B2B allowed identity providers (SAML IDPs through direct federation or Social Logins)
    - Conditional Access Policies changes
- Application and service principal objects:
    - New Applications / Service principals that might require Conditional Access policies
    - Application Consent activity
- Management group activity: The following Identity Aspects of management groups should be monitored:
    - RBAC role assignments at the MG
    - Azure Policies applied at the MG
    - Subscriptions moved between MGs
    - Any changes to security policies to the Root MG
- Custom roles:
    - Updates of the custom role definitions
    - New custom roles created
- Custom governance rules: If your organizations established any separation of duties rules (e.g., a holder of a Global Administrator tenant GA cannot be owner/contributor of subscriptions), create alerts or configure periodic reviews to detect violations.

Other Monitoring Considerations. Azure subscriptions that contain resources used for Log Management should be considered as critical infrastructure (Tier 0) and locked down to the Security Operations team of the corresponding environment. Consider using tools such as Azure Policy to enforce additional controls to these subscriptions.

Operational Tools

Cross-Environment Tooling Design

- Whenever possible, operational tools that will be used across multiple tenants should be designed to run as an Azure AD multi-tenant application to avoid re-deployment of multiple instances on each tenant and avoid operational inefficiencies. The implementation should include authorization logic in to ensure that isolation between users and data is preserved.
- Add alerts and detections to monitor any cross-environment automation (e.g., identity provisioning) and threshold limits for fail-safes. For example, you may want an alert if deprovisioning of user

accounts reaches a specific level, as it may indicate a bug or operational error that could have broad impact.

- Any automation that orchestrates cross-environment tasks should be operated as highly privileged system. This system should be homed to the highest security environment and pull from outside sources if data from other environments is required. Data validation and thresholds need to be applied to maintain system integrity. A common cross-environment task is identity lifecycle management to remove identities from all environments for a terminated employee.

IT Service Management Tools. Organizations using IT Service Management (ITSM) systems such as ServiceNow should configure Azure AD PIM role activation settings to request a ticket number as part of the activation purposes; organizations can also integrate Azure AD PIM APIs as part of existing service desk workflows if needed.

Similarly, Azure Monitor can be integrated with ITSM systems through the IT Service Management Connector.

Operational Practices. Minimize operational activities that require direct access to the environment to human identities. Instead model them as DevOps pipelines that execute common operations (e.g., add capacity to a PaaS solution, run diagnostics, etc.) and model direct access to the ARM interfaces to "break glass" scenarios.

Operations Challenges

- Activity of Service Principal Monitoring is limited for some scenarios
- Azure AD PIM alerts do not have an API. The mitigation is to have a regular review of those PIM alerts.
- Azure EA Portal does not provide monitoring capabilities. The mitigation is to have dedicated administration accounts and monitor the account activity.
- MCA does not provide audit logs for billing tasks. The mitigation is to have dedicated administration accounts and monitor the account activity.
- Some services in Azure needed to operate the environment need to be re-deployed and re-configured across environments as they can't be multi-tenant or multi-cloud.
- There is no full API coverage across Microsoft Online Services to fully achieve infrastructure as code. The mitigation is to leverage API's as much as possible and use portals for the remainder.
- There is no programmatic capability to discover resource tenants that have delegated subscription access to identities in a managing tenant. For example, if susie@fabrikam.com enabled a security group in the contoso.com tenant to manage subscriptions in the fabrikam.com tenant, administrators in the contoso.com do not have an API to discover that this delegation took place.
- Specific account activity monitoring (e.g., break-glass account, billing management account) is not provided out of the box. The mitigation is for customers to create their own alert rules.
- Tenant-wide configuration monitoring is not provided out of the box. The mitigation is for customers to create their own alert rules.

Conclusion

Azure AD provides an identity and access boundary for Azure resources and trusting applications. Most environment separation requirements can be fulfilled with delegated administration in a single Azure AD

tenant, and this configuration will reduce management overhead of your systems. However, some specific cases—for example complete resource and identity isolation—require multiple tenants.

You must determine your environment separation architecture based on your needs for the following:

- Resource separation. If a resource can change directory objects such as user objects, and this would interfere with other resources, the resource may need to be isolated in a multi-tenant architecture.
- Configuration separation. Tenant-wide configurations affect all resources. The effect of some tenant-wide configurations can be scoped with CA policies, and other methods. If you have a need for different tenant configurations that cannot be scoped with CA policies, you may need a multi-tenant architecture.
- Administrative separation. You can delegate the administration of management groups, subscriptions, resource groups, resources, and some policies within a single tenant. A Global Administrator always has access to everything within the tenant. If you need to ensure that the environment does not share administrators with another environment, you will need a multi-tenant architecture.

To stay secure, you must follow best practices for identity provisioning, authentication management, identity governance, lifecycle management, and operations consistently across all tenants.

Appendix

Azure Active Directory fundamentals

The following is a list of terms that are commonly associated with Azure AD and relevant to this white paper:

Terminology

Azure AD tenant. A dedicated and trusted instance of Azure AD that is automatically created when your organization signs up for a Microsoft cloud service subscription such as Microsoft Azure, Microsoft Intune, or Office 365. An Azure AD tenant generally represents a single organization or security boundary. The Azure AD tenant includes the users, groups, devices, and applications used to perform identity and access management for tenant resources.

Environment. In the context of this document, an environment is a collection of Azure subscriptions, Azure resources, and applications that are associated with one or more Azure AD tenets. The Azure AD tenant provides the identity control plane to govern access to these resources.

Production environment. In the context of this document, a production environment is the *live* environment with the infrastructure and services that end users directly interact with, for example a corporate or customer-facing environment.

Non-production environment. In the context of this document, a non-production environment refers to an environment used for development, testing, or lab purposes and is commonly referred to as a sandbox environment.

Identity. An identity is a directory object that can be authenticated and authorized for access to a resource. Identity objects exist for human identities, and non-human identities. Non-human entities include application objects, service principles, managed identities, and devices,

Human identities are user objects that generally represent people in an organization. These identities are either created and managed directly in Azure AD or are synchronized from an on-premises Active Directory to Azure AD for a given organization, which we will refer to as **local identities**. They can also be user objects invited from a partner organization or a social identity provider when using Azure AD B2B collaboration, which we will refer in this paper as **external identities.**

Non-human identities include application objects and service principals.

- **Application object**. An Azure AD application is defined by its one and only application object, which resides in the Azure AD tenant where the application was registered, known as the application's "home" tenant. **Single-tenant** applications are created to only authorize identities coming from the “home” tenant. Applications can also be created to be **multi-tenant** which will allow identities from any Azure AD tenant to authenticate.
- **Service principal object**. Although there are exceptions, application objects can be considered the definition of an application. Service principal objects can be considered an instance of an application. Service principals generally reference an application object, and one application object can be referenced by multiple service principals across directories.

Service principal objects are also directory identities that can perform tasks independently from human intervention. The service principal defines the access policy and permissions for a user or application in the Azure AD tenant. This enables core features such as authentication of the user or application during sign-in and authorization during resource access.

Azure AD allows application and service principal objects to authenticate with a password (also known as an application secret), or with a certificate. The use of passwords for service principals is discouraged and we recommend using a certificate whenever possible.

- **Managed identities for Azure resources.** Managed identities are special service principals in Azure AD that can be used to authenticate against services that support Azure AD authentication without needing to store credentials in your code or handle secrets management. For more information, see What are managed identities for Azure resources?
- **Device identity:** A device identity is an identity that verifies that the device being used in the authentication flow has undergone a process to attest that the device is legitimate and meets the technical requirements as specified by the organization. Once the device has successfully completed this process, the associated identity can be used to further control access to an organization's resources. Azure AD allows devices to authenticate with a certificate.

Some legacy scenarios require a "human identity" to be used in "non-human" scenarios. An example of this is service accounts being used in on-premises applications such as scripts or batch jobs that require access to Azure AD. This pattern is not recommended unless you protect your Azure AD accounts with Azure Multi-Factor Authentication.

Hybrid identity. A hybrid identity is an identity that spans on-premises and cloud environments. This provides the benefit of being able to use the same identity to access on-premises and cloud resources. The source of authority in this scenario is typically an on-premises directory, and the identity lifecycle around provisioning, de-provisioning and resource assignment is also generally driven from on-premises. For more information, see Hybrid identity documentation.

Directory objects. An Azure AD tenant contains the following common objects:

- **User objects** represent human identities and non-human identities for services that currently do not support service principals. User objects contain attributes that have the required information about the user including personal details, group memberships, devices, and roles assigned to the user.
- **Device objects** represent devices that are associated with an Azure AD tenant. Device objects contain attributes that have the required information about the device. This includes the operating system, associated user, compliance state, and the nature of the association with the Azure AD tenant. This association can take multiple forms depending on the nature of the interaction and trust level of the device.
    - **Hybrid Domain Joined**. Devices that are owned by the organization and joined to both the on-premises Active Directory and Azure AD. Typically a device purchased and managed by an organization and managed by System Center Configuration Manager.

- **Azure AD Domain Joined**. Devices that are owned by the organization and joined to the organization's Azure AD tenant. Typically a device purchased and managed by an organization which is joined to Azure AD and managed by a service such as Microsoft Intune.
- **Azure AD Registered**. Devices owned by the organization, or personal devices, used to access company resources. Usually a personal device used to access corporate resources. Organizations may require the device be enrolled via Mobile Device Management (MDM), or enforced through Mobile Application Management (MAM) without enrollment to access resources. This capability can be provided by a service such as Microsoft Intune.

- **Group objects** contain objects for the purposes of assigning resource access, applying controls, or configuration. Group objects contain attributes that have the required information about the group including the name, description, group members, group owners, and the group type. Groups in Azure AD take multiple forms based on an organization's requirements and can be mastered in Azure AD or synchronized from on-premises Active Directory Domain Services (AD DS).
  - **Assigned groups.** In Assigned groups users are added to or removed from the group on a manual basis, synchronized from on-premises AD DS, or updated as part of an automated scripted workflow. An assigned group can be synchronized from on-premises AD DS or homed in Azure AD.
  - **Dynamic membership groups.** In Dynamic groups users are assigned to the group automatically based on defined attributes. This allows group membership to be dynamically updated based on data held within the user objects. A dynamic group can only be homed in Azure AD.

Microsoft Account (MSA). You can create Azure subscriptions and tenants using Microsoft Accounts (MSA). A Microsoft Account is a personal account (as opposed to an organizational account) and is commonly used by developers and for trial scenarios. When used, the personal account is always made a guest in an Azure AD tenant.

Azure AD functional areas

These are the functional areas provided by Azure AD that are relevant to isolated environments. To learn more about the capabilities of Azure AD, see What is Azure Active Directory?.

Authentication

Authentication. Azure AD provides support for authentication protocols compliant with open standards such as Open ID Connect, OAuth and SAML. Azure AD also provides capabilities to allow organizations to federate existing on-premises identity providers such as Active Directory Federation Services (AD FS) to authenticate access to Azure AD integrated applications.

Azure AD provides industry leading strong authentication options that organizations can leverage to secure access to resources. Azure Multi-Factor Authentication, device authentication and password-less capabilities allow organizations to deploy strong authentication options that suit their workforce's requirements.

Single sign-on (SSO). With single sign-on, users sign in once with one account to access all resources that trust the directory such as domain-joined devices, company resources, software as a service (SaaS) applications, and all Azure AD integrated applications. For more information, see Single sign-on to applications in Azure Active Directory.

Authorization

Resource access assignment. Azure AD provides the ability to provide and secure access to resources. Assigning access to a resource in Azure AD can be done in two ways:

- **User assignment**: The user is directly assigned access to the resource and the appropriate role or permission is assigned to the user.
- **Group assignment**: A group containing one or more users is assigned to the resource and the appropriate role or permission is assigned to the group

Application access policies. Azure AD provides capabilities to further control and secure access to your organization's applications.

Conditional Access. Azure AD Conditional Access policies provide the ability to bring user and device context into the authorization flow when accessing Azure AD resources. Organizations should explore use of Conditional Access policies to allow, deny, or enhance authentication based on user, risk, device, and network context. For more information, see the Azure AD Conditional Access documentation.

Azure AD Identity Protection. This feature enables organizations to automate the detection and remediation of identity-based risks, investigate risks, and export risk detection data to third-party utilities for further analysis. See Azure AD Identity Protection for more information.

Administration

Identity management. Azure AD provides the ability to manage the lifecycle of user, group, and device identities. Azure AD Connect enables organizations to extend their current, on-premises identity management solution to the cloud. Azure AD Connect manages the provisioning, de-provisioning, and updates to these identities in Azure AD.

Azure AD also provides a portal and the Microsoft Graph API to allow organizations to manage identities or integrate Azure AD identity management into existing workflows or automation. To learn more about Microsoft Graph, see Use the Microsoft Graph API.

Device management. Azure AD provides the ability to manage the lifecycle and integration with cloud and on-premises device management infrastructures. It also provides the ability to define policies to control access from cloud or on-premises devices to your organizational data. Azure AD provides the lifecycle services of devices in the directory as well as the credential provisioning to enable authentication. It also manages a key attribute of a device in the system which is the level of trust. This is important when designing a resource access policy. For more information, see Azure AD Device Management documentation.

Configuration management. Azure AD has service elements that need to be configured and managed to ensure the service is configured to an organization's requirements. These elements include domain management, SSO configuration, and application management to name but a few. Azure AD provides a portal and the Microsoft Graph API to allow organizations to manage these elements or integrate into existing processes. To learn more about Microsoft Graph, see Use the Microsoft Graph API.

Governance

Identity lifecycle. Azure AD provides capabilities to create, retrieve, delete, and update identities in the directory, including external identities. Azure AD also provides services to automate the identity lifecycle to

ensure it is maintained in line with your organization's needs. For example, using Access Reviews to remove external users who have not signed in for a specified period.

Reporting and analytics. An important aspect of identity governance is visibility into user actions. Azure AD provides insights into your environment's security and usage patterns. These insights include detailed information on

- what your users access
- where they access it from
- the devices they use
- applications used to access

Azure AD also provides information on the actions that are being performed within Azure AD, and reports on security risks. For more information, see Azure Active Directory reports and monitoring.

Auditing. Auditing provides traceability through logs for all changes done by specific features within Azure AD. Examples of activities found in audit logs include changes made to any resources within Azure AD like adding or removing users, apps, groups, roles, and policies. Reporting in Azure AD enables you to audit sign-in activities, risky sign-ins, and users flagged for risk. For more information, see Audit activity reports in the Azure Active Directory portal.

Access certification. Access certification is the process to prove that a user is entitled to have access to a resource at a point in time. Azure AD Access Reviews continually review the memberships of groups or applications and provide insight to determine whether access is required or should be removed. This enables organizations to effectively manage group memberships, access to enterprise applications, and role assignments to make sure only the right people have continued access. For more information, see What are Azure AD access reviews?

Privileged access. Azure AD Privileged Identity Management (PIM) provides time-based and approval-based role activation to mitigate the risks of excessive, unnecessary, or misused access permissions to Azure resources. It is used to protect privileged accounts by lowering the exposure time of privileges and increasing visibility into their use through reports and alerts.

Self-service management

Credential registration. Azure AD provides capabilities to manage all aspects of user identity lifecycle and self-service capabilities to reduce the workload of an organization's helpdesk.

Group management. Azure AD provides capabilities that enable users to request membership in a group for resource access and to create groups that can be used for securing resources or collaboration. These capabilities can be controlled by the organization so that appropriate controls are put in place.

Consumer Identity and Access Management (IAM)

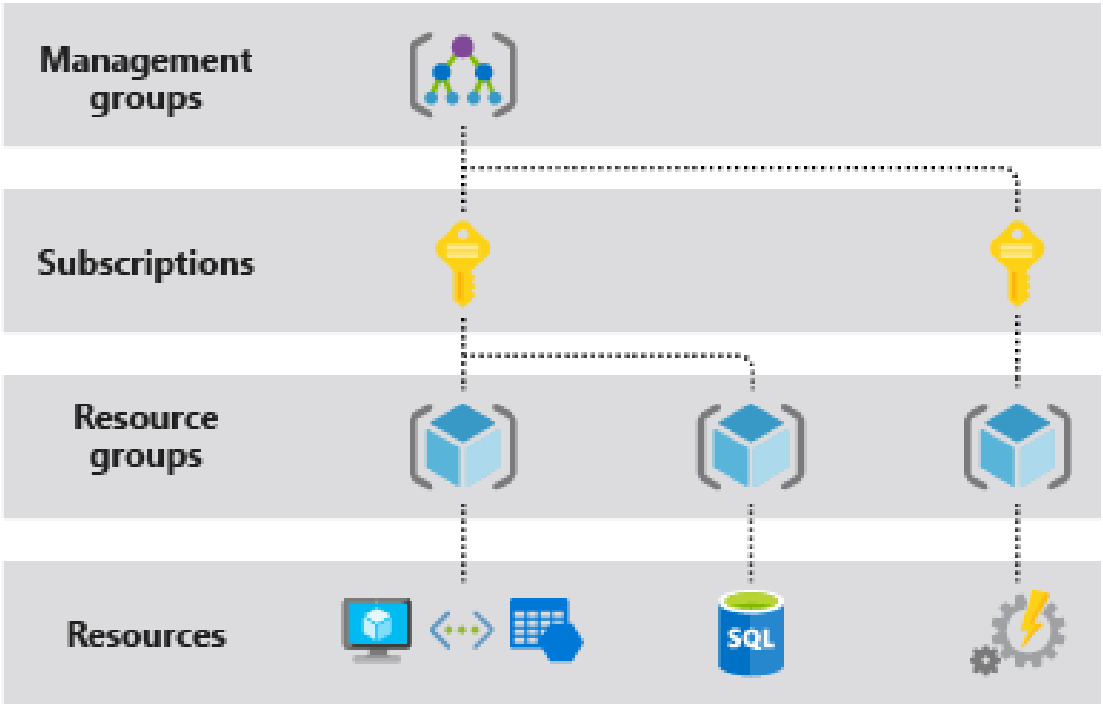
Azure AD B2C. Azure AD B2C is a service that can be enabled in an Azure subscription to provide identities to consumers for your organization's customer-facing applications. This is a separate island of identity and these users do not appear in the organization's Azure AD tenant. Azure AD B2C is managed by administrators in the tenant associated with the Azure subscription.

Azure resource management fundamentals

Now that you have been introduced to concepts specific to Azure AD, it is equally important to understand some terms that are specific to Azure resources.

Terminology

The following image shows an example of the four levels of scope that are provided by Azure:



Azure resource management model

Resource. A manageable item that is available through Azure. Virtual machines, storage accounts, web apps, databases, and virtual networks are examples of resources.

Resource group. A container that holds related resources for an Azure solution such as a collection of virtual machines, associated VNets, and load balancers that require management by specific teams. The resource group includes those resources that you want to manage as a group. You decide which resources belong in a resource group based on what makes the most sense for your organization.

Subscription. From an organizational hierarchy perspective, a subscription is a billing and management container of resources and resource groups. An Azure subscription has a trust relationship with Azure AD. A subscription trusts Azure AD to authenticate users, services, and devices. Note, a subscription may trust only one Azure AD tenant. However, each tenant may trust multiple subscriptions.

Management group. Azure management groups provide a hierarchical method of applying policies and compliance at different scopes above subscriptions. It can be at the tenant root management group (highest scope) or at lower levels in the hierarchy. You organize subscriptions into containers called "management groups" and apply your governance conditions to the management groups. All subscriptions within a management group automatically inherit the conditions applied to the management group. Note, policy definitions can be applied to a management group or subscription.

Resource provider. A service that supplies Azure resources. For example, a common resource provider is Microsoft.Compute, which supplies the virtual machine resource. Microsoft.Storage is another common resource provider.

Resource Manager template. A JavaScript Object Notation (JSON) file that defines one or more resources to deploy to a resource group, subscription, tenant, or management group. The template can be used to deploy the resources consistently and repeatedly. See Template deployment overview.

Azure Resource Management Model

Each Azure subscription is associated with controls used by Azure Resource Manager (ARM). ARM is the deployment and management service for Azure, it has a trust relationship with Azure AD for identity management for organizations, and the Microsoft Account (MSA) for individuals. ARM provides a management layer that enables you to create, update, and delete resources in your Azure subscription. You use management features like access control, locks, and tags, to secure and organize your resources after deployment.

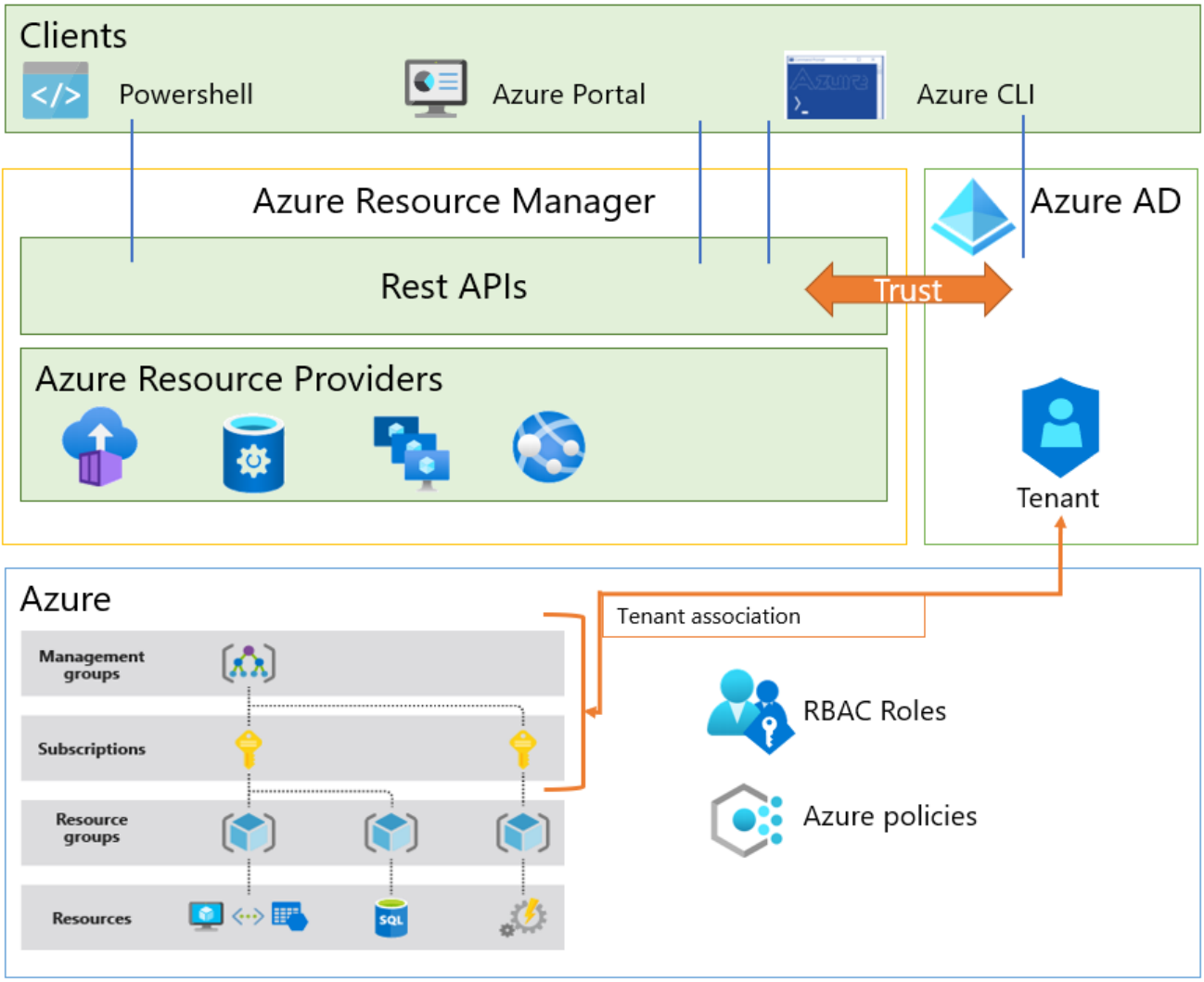
Note: Prior to ARM, there was another deployment model called Azure Service Manager (ASM) or "classic". To learn more, see Azure Resource Manager vs. classic deployment. Managing environments with the ASM model is out of scope of this document.

Azure Resource Manager is the front-end service which hosts the REST APIs used by PowerShell, the Azure Portal, or other clients to manage resources. When a client makes a request to manage a specific resource, ARM proxies the request to the resource provider to complete the request. For example, if a client makes a request to manage a virtual machine resource, ARM proxies the request to the Microsoft.Compute resource provider. ARM requires the client to specify an identifier for both the subscription and the resource group to manage the virtual machine resource.

Before any resource management request can be executed by ARM, a set of controls are checked.

- **Valid user check.** The user requesting to manage the resource must have an account in the Azure AD tenant associated with the subscription of the managed resource.
- **User permission check.** Permissions are assigned to users using role-based access control (RBAC). An RBAC role specifies a set of permissions a user may take on a specific resource. RBAC helps you manage who has access to Azure resources, what they can do with those resources, and what areas they have access to.
- **Azure policy check.** Azure policies specify the operations allowed or explicitly denied for a specific resource. For example, a policy can specify that users are only allowed (or not allowed) to deploy a specific type of virtual machine.

The following diagram summarizes the resource model we just described.



Azure resource management with ARM and Azure AD

Azure Lighthouse. Azure Lighthouse enables resource management across tenants. Organizations can delegate roles at the subscription or resource group level to identities in another tenant.

Subscriptions that enable delegated resource management with Azure Lighthouse have attributes that indicate the tenant IDs that can manage subscriptions or resource groups, as well as mapping between the built-in RBAC role in the resource tenant to identities in the service provider tenant. At runtime, Azure ARM will consume these attributes to authorize tokens coming from the service provider tenant.

It is worth noting that Azure Lighthouse itself is modeled as an Azure resource provider, which means that aspects of the delegation across a tenant can be targeted through Azure Policies.

Azure resource management with Azure AD

Now that you have a better understanding of the resource management model in Azure, let's briefly examine some of the capabilities of Azure AD that can provide identity and access management for Azure resources.

Billing

Billing is important to resource management because some billing roles interact with or can manage resources. Billing works differently depending on the type of agreement that you have with Microsoft.

Azure Enterprise Agreements

Azure Enterprise Agreement (Azure EA) customers are onboarded to the Azure EA Portal upon execution of their commercial contract with Microsoft. Upon onboarding an identity is associated to a "root" Enterprise Administrator billing role. The portal provides a hierarchy of management functions:

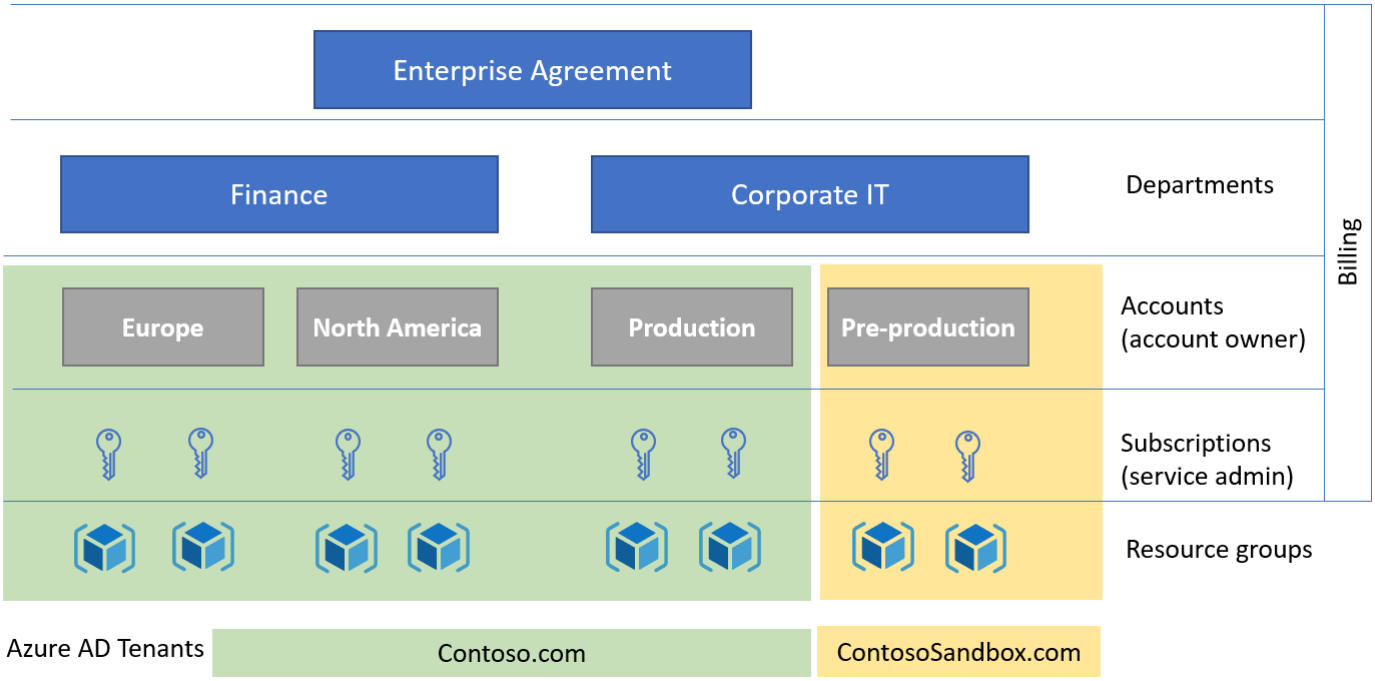
- Departments help you segment costs into logical groupings and enable you to set a budget or quota at the department level.
- Accounts are used to further segment departments. You can use accounts to manage subscriptions and to access reports.
  The EA portal can authorize Microsoft Accounts (MSA) or Azure AD accounts (identified in the portal as "Work or School Accounts"). Identities with the role of "Account Owner" in the EA portal can create Azure subscriptions.

Enterprise billing and Azure AD tenants

When an Account Owner creates an Azure subscription within an enterprise agreement, the identity and access management of the subscription is configured as follows:

- The Azure subscription is associated with the same Azure AD tenant of the Account Owner.
- The account owner who created the subscription will be assigned the Service Administrator and Account Administrator roles. (The Azure EA Portal assigns Azure Service Manager (ASM) or "classic" roles to manage subscriptions. To learn more, see Azure Resource Manager vs. classic deployment.)

An enterprise agreement can be configured to support multiple tenants by setting the authentication type of "Work or school account cross-tenant" in the Azure EA Portal. Given the above, organizations can set multiple accounts for each tenant, and multiple subscriptions for each account, as shown in the diagram below.



Enterprise Agreement billing structure

It is important to note that the default configuration described above grants the Azure EA Account Owner privileges to manage the resources in any subscriptions they created. For subscriptions holding production workloads, consider decoupling billing and resource management by changing the service administrator of the subscription right after creation.

To further decouple and prevent the account owner from regaining service administrator access to the subscription, the subscription's tenant can be changed after creation. If the account owner does not have a user object in the Azure AD tenant the subscription is moved to, they cannot regain the service owner role.

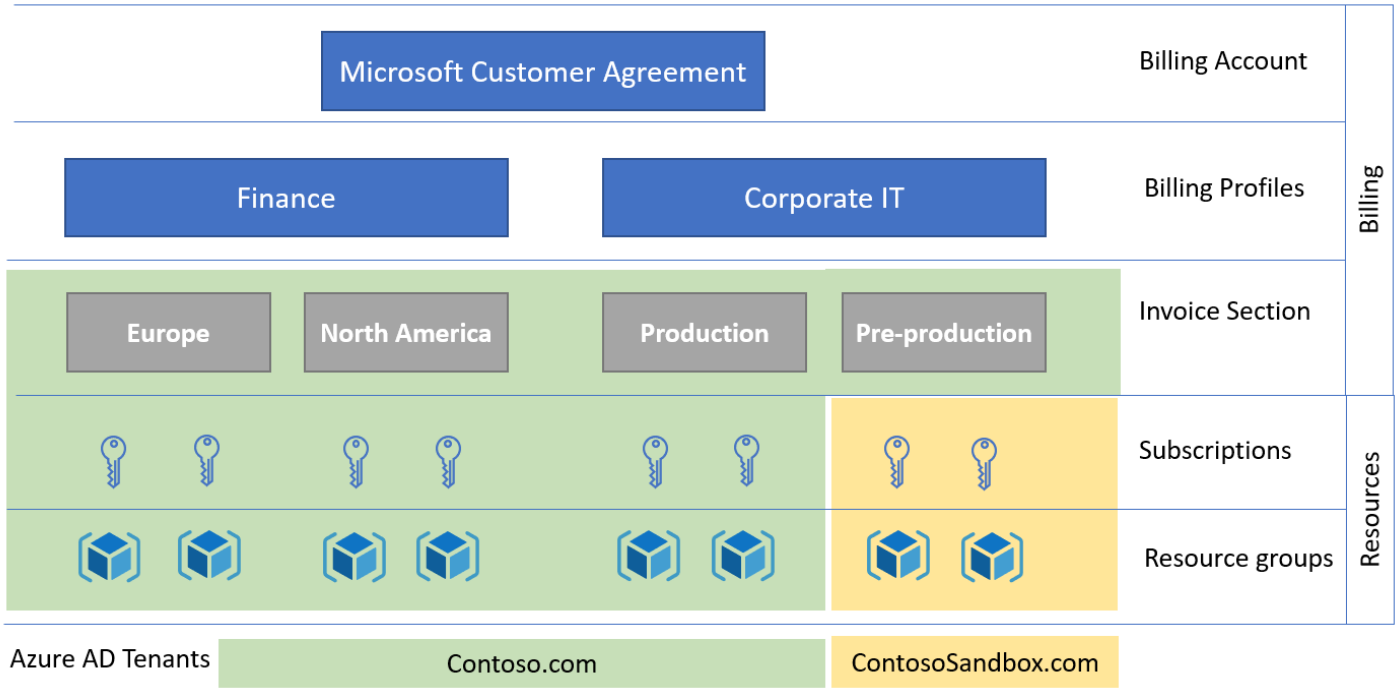
To learn more, visit Classic subscription administrator roles, Azure RBAC roles, and Azure AD roles.

Microsoft Customer Agreement

Customers enrolled with a Microsoft Customer Agreement (MCA) have a different billing management system with its own roles.

A billing account for the Microsoft Customer Agreement contains one or more billing profiles that allow managing invoices and payment methods. Each billing profile contains one or more invoice sections to organize costs on the billing profile's invoice.

In a Microsoft Customer Agreement, billing roles come from a *single Azure AD tenant*. To provision subscriptions for multiple tenants, the subscriptions must be initially created in the same Azure AD Tenant as the MCA, and then changed. In the diagram below, the subscriptions for the Corporate IT pre-production environment were moved to the ContosoSandbox tenant after creation.



MCA billing structure

RBAC and role assignments in Azure

In the Azure Resource Management Fundamentals section, you learned Azure RBAC is the authorization system that provides fine-grained access management to Azure resources, and includes many built-in roles. You can create custom roles, and assign roles at different scopes. Permissions are enforced by assigning RBAC roles to objects requesting access to Azure resources.?

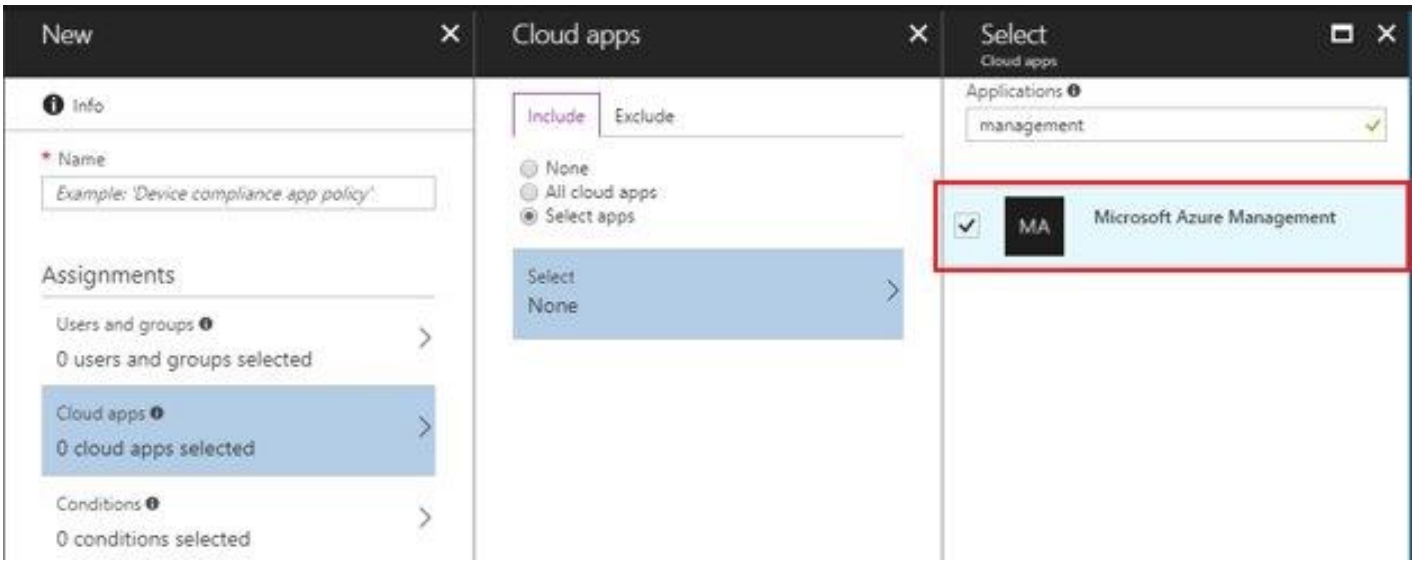
Azure AD roles operate on concepts similar to Azure role-based access control. The difference between these two role-based access control systems is that Azure RBAC uses Azure Resource Management to control access to Azure resources such as virtual machines or storage, and Azure AD roles control access to Azure AD, applications, and Microsoft services such as Office 365.

Both Azure AD roles and Azure RBAC roles integrate with Azure AD Privileged Identity Management to enable just-in-time activation policies such as approval workflow and MFA.

Conditional Access

Azure AD Conditional Access (CA) can be used to manage access to Azure management endpoints. CA policies can be applied to the Microsoft Azure Management cloud app to protect the Azure resource management endpoints such as:

- Azure Resource Manager Provider (services)
- Azure Resource Manager APIs
- Azure PowerShell
- Azure CLI
- Azure Portal



Conditioanl Access policy

For example, an administrator may configure a Conditional Access policy which allows a user to sign into the Azure Portal only from approved locations and also requires either Azure Multi-Factor Authentication or a hybrid Azure AD domain-joined device.

Azure Managed Identities

A common challenge when building cloud applications is how to manage the credentials in your code for authenticating to cloud services. Keeping the credentials secure is an important task. Ideally, the credentials never appear on developer workstations and aren't checked into source control. Managed identities for Azure resources provide Azure services with an automatically managed identity in Azure AD. You can use the identity to authenticate to any service that supports Azure AD authentication without any credentials in your code.

There are two types of managed identities:

- A system-assigned managed identity is enabled directly on an Azure resource. When the resource is enabled, Azure creates an identity for the resource in the associated subscriptions's trusted Azure AD tenant. After the identity is created, the credentials are provisioned onto the resource. The lifecycle of a system-assigned identity is directly tied to the Azure resource. If the resource is deleted, Azure automatically cleans up the credentials and the identity in Azure AD.
- A user-assigned managed identity is created as a standalone Azure resource. Azure creates an identity in the Azure AD tenant that's trusted by the subscriptioin with which the resource is assocated. After the identity is created, the identity can be assigned to one or more Azure resources. The lifecycle of a user-assigned identity is managed separately from the lifecycle of the Azure resources to which it's assigned.

Internally, managed identities are service principals of a special type, to only be used by specific Azure resources. When the managed identity is deleted, the corresponding service principal is automatically removed.

Azure Active Directory Domain Services

Azure Active Directory Domain Services (Azure AD DS) provides a managed domain to facilitate authentication for Azure workloads using legacy protocols. Supported servers are moved from an on-premises AD DS forest and joined to an Azure AD DS managed domain and continue to use legacy protocols for authentication (e.g., Kerberos authentication).

Azure AD B2C directories and Azure

An Azure AD B2C tenant is linked to an Azure subscription for billing and communication purposes. Azure AD B2C tenants have a self-contained role structure in the directory which is independent from the Azure RBAC privileged roles of the Azure subscription..

When the Azure AD B2C tenant is initially provisioned, the user creating the B2C tenant must have contributor or owner permissions in the subscription. Upon creation, that user becomes the first Azure AD B2C tenant global administrator and they can subsequently create other accounts and assign them to directory roles.

It is important to note that the owners and contributors of the linked Azure AD subscription can remove the link between the subscription and the directory, which will affect the ongoing billing of the Azure AD B2C usage.

Identity Considerations for Infrastructure as a Service Solutions in Azure

This scenario addresses identity isolation requirements that organizations have for Infrastructure as a Service (IaaS) workloads.

There are three key options regarding isolation management of IaaS workloads:

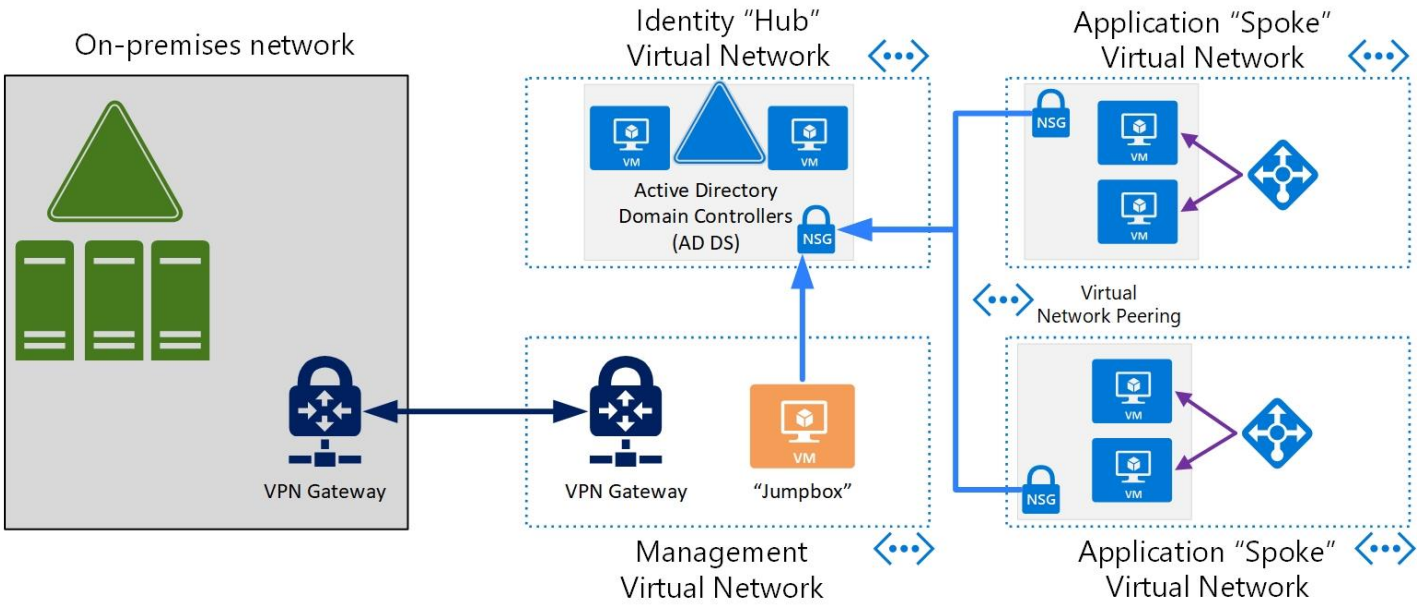
- Virtual machines joined to stand-alone Active Directory Domain Services (AD DS)
- Azure Active Directory Domain Services (Azure AD DS) joined virtual machines
- Sign-in to virtual machines in Azure using Azure AD authentication

A key concept to address with the first two options is that there are two identity realms that are involved in these scenarios.

- When you log on to an Azure Windows Server VM via remote desktop protocol (RDP), you are generally logging onto the server using your domain credentials which performs a Kerberos authentication against an on-premises AD DS domain controller or Azure AD DS. Alternatively, if the server is not domain-joined then a local account can be used to log on to the virtual machines.
- When you sign into the Azure Portal to create or manage a VM you are authenticating against Azure AD (potentially using the same credentials if you have synchronized the correct accounts) and this could result in an authentication against your domain controllers should you be using Active Directory Federation Services (AD FS) or PassThrough Authentication.

Virtual machines joined to standalone Active Directory Domain Services

AD DS is the Windows Server based directory service that organizations have largely adopted for on-premises identity services. AD DS can be deployed when a requirement exists to deploy IaaS workloads to Azure that require identity isolation from AD DS administrators and users in another forest.



AD DS virtual machine management

The following considerations need to be made in this scenario:

AD DS domain controllers: a minimum of two AD DS domain controllers must be deployed to ensure that authentication services are highly available and performant. For more information, see AD DS Design and Planning.

AD DS Design and Planning: A new AD DS forest must be created with the following services configured correctly:

- AD DS Domain Name Services (DNS). AD DS DNS must be configured for the relevant zones within AD DS to ensure that name resolution operates correctly for servers and applications.
- AD DS Sites and Services. These services must be configured to ensure that applications have low latency and performant access to domain controllers. The relevant virtual networks, subnets, and data center locations that servers are located in should be configured in sites and services.
- AD DS FSMOs. The Flexible Single Master Operation (FSMO) roles that are required should be reviewed and assigned to the appropriate AD DS domain controllers.
- AD DS Domain Join. All servers (excluding "jumpboxes") that require AD DS for authentication, configuration and management need to be joined to the isolated forest.
- AD DS Group Policy (GPO). AD DS GPOs must be configured to ensure that the configuration meets the security requirements and that the configuration is standardized across the forest and domain-joined machines.
- AD DS Organizational Units (OU): AD DS OUs must be defined to ensure grouping of AD DS resources into logical management and configuration silos for purposes of administration and application of configuration.

- Role-based access control. RBAC must be defined for administration and access to resources joined to this forest. This includes:
    - AD DS Groups. Groups must be created to apply appropriate permissions for users to AD DS resources.
    - Administration accounts. As mentioned at the start of this section there are two administration accounts required to manage this solution.
        - An AD DS administration account with the least privileged access required to perform the administration required in AD DS and domain-joined servers.
        - An Azure AD administration account for Azure Portal access to connect, manage, and configure virtual machines, VNets, network security groups and other required Azure resources.
    - AD DS user accounts. Relevant user accounts need to be provisioned and added to correct groups to allow user access to applications hosted by this solution.

Virtual networks (VNets)

- AD DS domain controller IP address: The domain controllers should not be configured with static IP addresses within the operating system. The IP addresses should be reserved on the Azure VNet to ensure they always stay the same and DC should be configured to use DHCP.
- VNet DNS Server. DNS servers must be configured on VNets that are part of this isolated solution to point to the domain controllers. This is required to ensure that applications and servers can resolve the required AD DS services or other services joined to the AD DS forest.
- Network security groups (NSGs). The domain controllers should be located on their own VNet or subnet with NSGs defined to only allow access to domain controllers from required servers (e.g., domain-joined machines or jumpboxes).

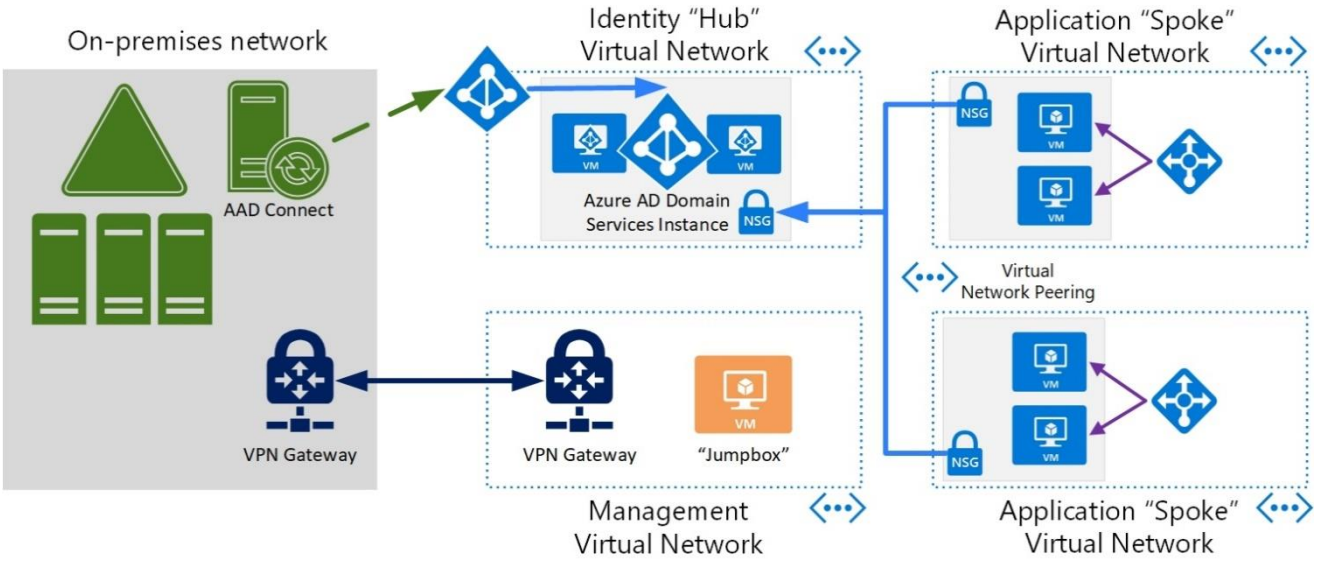
Challenges: The list below highlights key challenges with using this option for identity isolation:

- An additional AD DS Forest to administer, manage and monitor resulting in more work for the IT team to perform.
- Further infrastructure may be required for management of patching and software deployments. Organizations should consider deploying Azure Update Management, Group Policy (GPO) or System Centre Configuration Manager (SCCM) to manage these servers.
- Additional credentials for users to remember and use to access resources.

For this isolated model, it is assumed that there is no connectivity to or from the domain controllers from the customer's corporate network and that there are no trusts configured with other forests. A jumpbox or management server should be created to allow a point from which the AD DS domain controllers can be managed and administered.

Azure Active Directory Domain Services joined virtual machines

When a requirement exists to deploy IaaS workloads to Azure that require identity isolation from AD DS administrators and users in another forest, then an Azure AD Domain Services (Azure AD DS) managed domain can be deployed. Azure AD DS is a service that provides a managed domain to facilitate authentication for Azure workloads using legacy protocols. This provides an isolated domain without the technical complexities of building and managing your own AD DS. The following considerations need to be made.



Azure AD DS virtual machine management

Azure AD DS managed domain: Only one Azure AD DS managed domain can be deployed per Azure AD tenant and this is bound to a single VNet. It is recommended that this VNet forms the "hub" for Azure AD DS authentication. From this hub, "spokes" can be created and linked to allow legacy authentication for servers and applications. The spokes are additional VNets on which Azure AD DS joined servers are located and are linked to the hub using Azure network gateways or VNet peering.

User forest vs. resource forest: Azure AD DS provides two options for forest configuration of the Azure AD DS managed domain. For the purposes of this section we focus on user forest, as the resource forest relies on a trust being configured with an AD DS forest and this goes against the isolation principal we are addressing here.

- **User forest**: By default, an Azure AD DS managed domain is created as a user forest. This type of forest synchronizes all objects from Azure AD, including any user accounts synchronized from an on-premises AD DS environment.
- **Resource forest**: Resource forests only synchronize users and groups created directly in Azure AD and requires a trust be configured with an AD DS forest for user authentication. For more information, see Resource forest concepts and features for Azure Active Directory Domain Services.

Managed domain location: A location must be set when deploying an Azure AD DS managed domain. The location is a physical region (data center) where the managed domain is deployed. It is recommended you:

- Consider a location that is geographically close to the servers and applications that require Azure AD DS services.
- Consider regions that provide Availability Zones capabilities for high availability requirements. For more information, see Regions and Availability Zones in Azure.

Object provisioning: Azure AD DS synchronizes identities from the Azure AD that is associated with the subscription that Azure AD DS is deployed into. It is also worth noting that if the associated Azure AD has synchronization set up with Azure AD Connect (user forest scenario) then the life cycle of these identities can also be reflected in Azure AD DS. This service has two modes that can be used for provisioning user and group objects from Azure AD.

- **All:** All users and groups are synchronized from Azure AD into Azure AD DS.
- **Scoped:** Only users in scope of a group(s) are synchronized from Azure AD into Azure AD DS.

When you first deploy Azure AD DS, an automatic one-way synchronization is configured to replicate the objects from Azure AD. This one-way synchronization continues to run in the background to keep the Azure AD DS managed domain up to date with any changes from Azure AD. No synchronization occurs from Azure AD DS back to Azure AD. For more information, see How objects and credentials are synchronized in an Azure AD Domain Services managed domain.

It is worth noting that if you need to change the type of synchronization from All to Scoped (or vice versa), then the Azure AD DS managed domain will need to be deleted, recreated and configured. In addition, organizations should consider the use of "scoped" provisioning to reduce the identities to only those that need access to Azure AD DS resources as a good practice.

Group Policy Objects (GPO): To configure GPO in an Azure AD DS managed domain you must use Group Policy Management tools on a server that has been domain joined to the Azure AD DS managed domain. For more information, see Administer Group Policy in an Azure AD Domain Services managed domain.

Secure LDAP: Azure AD DS provides a secure LDAP service that can be used by applications that require it. This setting is disabled by default and to enable secure LDAP a certificate needs to be uploaded, in addition the NSG that secures the VNet that Azure AD DS is deployed on to must allow port 636 connectivity to the Azure AD DS managed domains. For more information, see Configure secure LDAP for an Azure Active Directory Domain Services managed domain.

Administration: To perform administration duties on Azure AD DS (e.g., domain join machines or edit GPO), the account used for this task needs to be part of the AAD DC Administrators group. Accounts that are members of this group cannot directly sign-in to domain controllers to perform management tasks. Instead, you create a management VM that is joined to the Azure AD DS managed domain, then install your regular AD DS management tools. For more information, see Management concepts for user accounts, passwords, and administration in Azure Active Directory Domain Services.

Password hashes: For authentication with Azure AD DS to work, password hashes for all users need to be in a format that is suitable for NT LAN Manager (NTLM) and Kerberos authentication. To ensure authentication with Azure AD DS works as expected, the following prerequisites need to be performed.

- **Users synchronized with Azure AD Connect (from AD DS)**: The legacy password hashes need to be synchronized from on-premises AD DS to Azure AD.

- **Users created in Azure AD:** Need to reset their password for the correct hashes to be generated for usage with Azure AD DS. For more information, see Enable synchronization of password hashes.

Network: Azure AD DS is deployed on to an Azure VNet so considerations need to be made to ensure that servers and applications are secured and can access the managed domain correctly. For more information, see Virtual network design considerations and configuration options for Azure AD Domain Services.

- **Azure AD DS must be deployed in its own subnet**: Don't use an existing subnet or a gateway subnet.
- **A network security group (NSG)**: is created during the deployment of an Azure AD DS managed domain. This network security group contains the required rules for correct service communication. Do not create or use an existing network security group with your own custom rules.
- **Azure AD DS requires 3-5 IP addresses**: Make sure that your subnet IP address range can provide this number of addresses. Restricting the available IP addresses can prevent Azure AD DS from maintaining two domain controllers.
- **VNet DNS Server**: As previously discussed with regards to the "hub and spoke" model, it's important to have DNS configured correctly on the VNets to ensure that servers joined to the Azure AD DS managed domain have the correct DNS settings to resolve the Azure AD DS managed domain. Each VNet has a DNS server entry that is passed to servers as they obtain an IP address and these DNS entries need to be the IP addresses of the Azure AD DS managed domain. For more information, see Update DNS settings for the Azure virtual network.

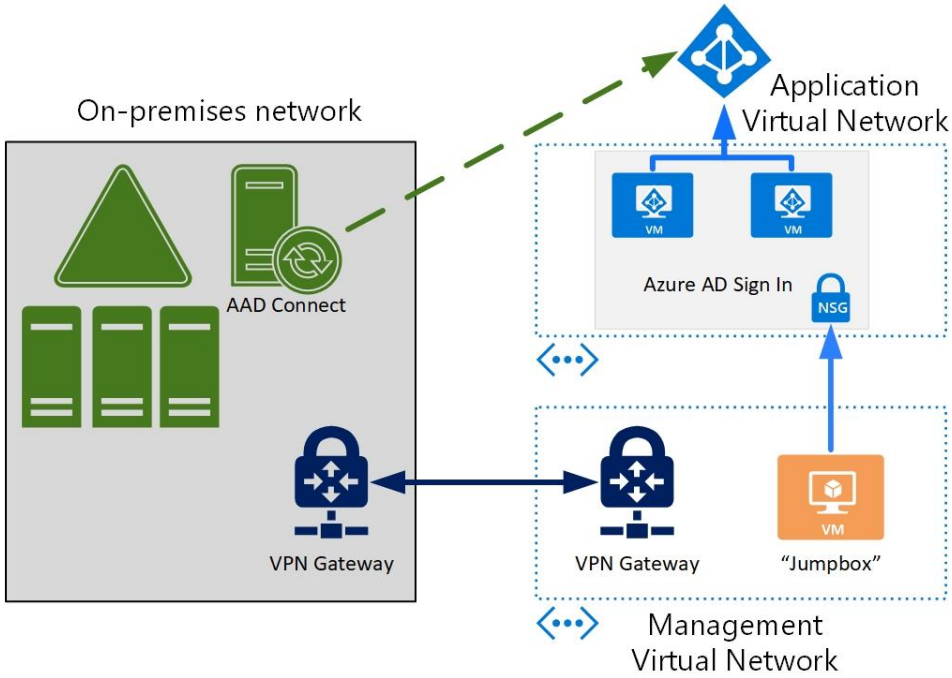
Challenges: The list below highlights key challenges with using this option for Identity Isolation.

- Some Azure AD DS configuration can only be administered from an Azure AD DS joined server.
- Only one Azure AD DS managed domain can be deployed per Azure AD tenant. As we describe in this section the hub and spoke model is recommended to provide Azure AD DS authentication to services on other VNets.
- Further infrastructure maybe required for management of patching and software deployments. Organizations should consider deploying Azure Update Management, Group Policy (GPO) or System Centre Configuration Manager (SCCM) to manage these servers.

For this isolated model, it is assumed that there is no connectivity to the VNet that hosts the Azure AD DS managed domain from the customer's corporate network and that there are no trusts configured with other forests. A jumpbox or management server should be created to allow a point from which the Azure AD DS can be managed and administered.

Sign into virtual machines in Azure using Azure Active Directory authentication

When a requirement exists to deploy IaaS workloads to Azure that require identity isolation, then the final option is to use Azure AD for logon to servers in this scenario. This provides the ability to make Azure AD the identity realm for authentication purposes and identity isolation can be achieved by provisioning the servers into the relevant subscription which is linked to the required Azure AD tenant. The following considerations need to be made.



Azure AD Authentication to Azure VMs.

Supported operating systems: Signing into virtual machines in Azure using Azure AD authentication is currently supported in Windows and Linux. For more specifics on supported operating systems, refer to the documentation for Windows and Linux.

Credentials: One of the key benefits of signing in to virtual machines in Azure using Azure AD authentication is the ability to use the same federated or managed Azure AD credentials that you normally use for access to Azure AD services for sign in to the virtual machine.

> **Note**: The Azure AD tenant that is used for sign-in in this scenario is the Azure AD tenant that is associated with the subscription that the virtual machine has been provisioned into. This Azure AD tenant can be one that has identities synchronized from on-premises AD DS. Organizations should make an informed choice that aligns with their isolation principals when choosing which subscription and Azure AD tenant they wish to use for sign-in to these servers.

Network Requirements: These virtual machines will need to access Azure AD for authentication so you must ensure that the virtual machines network configuration permits outbound access to Azure AD endpoints on 443. See the documentation for Windows and Linux for more information.

Role-based Access Control (RBAC): Two RBAC roles are available to provide the appropriate level of access to these virtual machines. These RBAC roles can be configured via the Azure AD Portal or via the Azure Cloud Shell Experience. For more information, see Configure role assignments for the VM.

- **Virtual machine administrator logon:** Users with this role assigned to them can log into an Azure virtual machine with administrator privileges.
- **Virtual machine user logon:** Users with this role assigned to them can log into an Azure virtual machine with regular user privileges.

Conditional Access: A key benefit of using Azure AD for signing into Azure virtual machines is the ability to enforce Conditional Access as part of the sign-in process. This provides the ability for organizations to require conditions to be met before allowing access to the virtual machine and to use Azure Multi-Factor Authentication to provide strong authentication. For more information, see Using Conditional Access.

> **Note**: Remote connection to virtual machines joined to Azure AD is only allowed from Windows 10 PCs that are Azure AD joined or hybrid Azure AD joined to the same directory as the virtual machine.

Challenges: The list below highlights key challenges with using this option for identity isolation.

- No central management or configuration of servers. i.e., no Group Policy that can be applied to a group of servers. Organizations should consider deploying Update Management in Azure to manage patching and updates of these servers.
- Not suitable for multi-tiered applications that have requirements to authenticate with on-premises mechanisms such as Windows Integrated Authentication across these servers or services. If this is a requirement for the organization, then it is recommended that you explore the Standalone Active Directory Domain Services, or the Azure Active Directory Domain Services scenarios described in this section.

For this isolated model, it is assumed that there is no connectivity to the VNet that hosts the virtual machines from the customer's corporate network. A jumpbox or management server should be created to allow a point from which these serves can be managed and administered.