



AZ-500 Domain 1: Secure Identity and Access

AZ-500 skills measured, domain 1 (15-20 %), as of 2026 | Entra roles vs Azure RBAC, scopes, custom roles, conditions, PIM, managed identities, service principals, app permissions, MFA recall

AZ-500 Domain 1 Entra ID Azure RBAC PIM

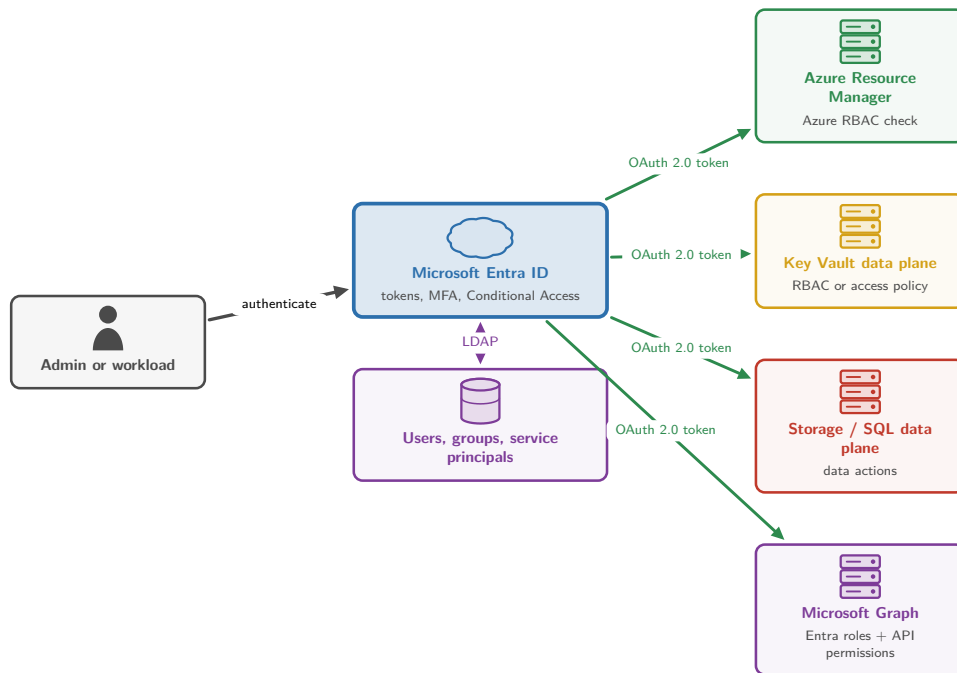
Two role systems, one tenant

Microsoft Entra roles

control the directory: users, groups, apps, Conditional Access, M365 admin
scope: tenant, or an administrative unit, or one app registration
examples: Global Administrator, Privileged Role Administrator, Security Administrator, Application Administrator
managed in the Entra admin center or Graph

Azure RBAC roles

control Azure resources through Azure Resource Manager (ARM)
scope: management group → subscription → resource group → resource
examples: Owner, Contributor, Reader, User Access Administrator, Role Based Access Control Administrator
managed in the Azure portal, az role, New-AzRoleAssignment, Bicep



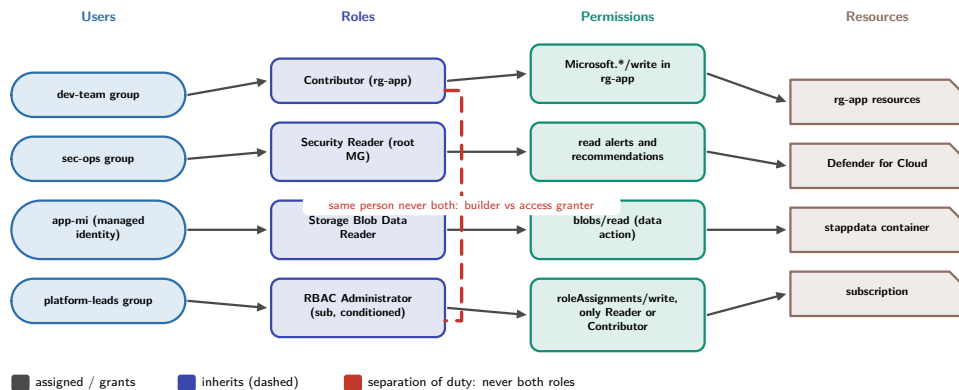
Every Azure plane trusts Entra tokens; what the token allows is decided by the receiving plane

EXAM TIP

A **Global Administrator** has no **Azure resource rights by default**. To regain control of all subscriptions, they turn on **Elevate access** (Entra ID → Properties → Access management for Azure resources), which assigns User Access Administrator at root /; remove it when done. Trap: 'give the security team read access to every subscription' → Security Reader (or Reader) at the **tenant root management group**, not a Global Admin role.

Azure RBAC: scopes, roles, conditions

Role	Can do	Cannot do
Owner	everything, including role assignments	-
Contributor	create and manage all resources	assign roles or create locks (Microsoft.Authorization/*/* write is in its NotActions), assign Blueprints
Reader	view resources (control plane)	read data (blobs, secrets) unless a data role is added
User Access Administrator	manage role assignments	manage resources
Role Based Access Control Administrator	assign roles, optionally limited by conditions (which roles, which principals)	manage resources; it cannot assign roles outside its condition
Security Admin / Security Reader	Defender for Cloud policies, alerts, recommendations / view them	change unrelated resources
Key Vault Secrets User / Officer	read secrets / manage secrets (data plane)	manage the vault itself (control plane)



Assign roles to groups or managed identities, at the narrowest scope; separate who builds from who grants

CUMULATIVE: assignments are inherited and add up down the tree; a deny blocks whatever is granted



Roles assigned higher up cannot be removed lower down; only a deny assignment or deny policy blocks them.

Azure RBAC is cumulative: each scope adds to what it inherits; only deny assignments subtract

- **Effective permissions** union of all role assignments on the scope and its parents (inherited downwards), minus deny assignments; Actions - NotActions is not a deny (another role can still grant it)
- **Control vs data plane** Actions = ARM operations (create a storage account); DataActions = data operations (read a blob, get a secret). Owner does not read blobs with Entra auth unless a data role is assigned
- **Conditions (ABAC)** on Storage Blob / Queue data roles: @Resource[...tags: Project] StringEquals @Principal[...]; delegated-assignment conditions restrict which roles an RBAC Administrator may hand out
- **Limits (as of 2026)** 4,000 role assignments per subscription, 5,000 custom roles per tenant; assign to groups to stay far below
- **Resource locks** CanNotDelete or ReadOnly, inherited, apply even to Owners; ReadOnly can break operations that use POST (list keys)

Custom role: VM operator who cannot delete (role.json)

```
{
  "Name": "VM Operator (no delete)",
  "IsCustom": true,
  "Description": "Start, stop, restart VMs; read the rest",
  "Actions": [
    "Microsoft.Compute/virtualMachines/read",
    "Microsoft.Compute/virtualMachines/start/action",
    "Microsoft.Compute/virtualMachines/restart/action",
    "Microsoft.Compute/virtualMachines/deallocate/action",
    "Microsoft.Resources/subscriptions/resourceGroups/read"
  ],
  "NotActions": [],
  "DataActions": [],
  "NotDataActions": [],
  "AssignableScopes": ["/subscriptions/<sub-id>"]
}
```

- 0 \$ az role assignment create
- 1 \$ az role assignment create --assignee-object-id <group-oid>
the principal: a group object ID (avoid UPN lookups in pipelines)
- 2 \$ az role assignment create --assignee-object-id <group-oid> --assignee-principal-type Group
skips the Graph lookup; required for just-created principals
- 3 \$ az role assignment create --assignee-object-id <group-oid> --assignee-principal-type Group --role "VM Operator (no delete)"
built-in or custom role name or ID
- 4 \$ az role assignment create --assignee-object-id <group-oid> --assignee-principal-type Group --role "VM Operator (no delete)" --scope /subscriptions/<sub>/resourceGroups/rg-app_
the narrowest scope that works

Worked example: why assign to groups

120 engineers × 3 roles × 10 resource groups assigned one by one = 120 × 3 × 10 = **3,600 assignments**, close to the 4,000-per-subscription limit and impossible to review. With 3 Entra groups (one per role) on the same 10 resource groups: 3 × 10 = **30 assignments**, and joiners/leavers change group membership only. Saving: 1 - 30 / 3,600 = 99 % fewer assignments to audit.

Privileged Identity Management (PIM)

PIM setting	What it does	Typical value
Eligible vs active	eligible = must activate; active = always on (permanent or time-bound)	eligible for every privileged role
Activation maximum duration	how long rights last after activation (1-24 h)	1-4 h for Owner / Global Admin
On activation, require	Azure MFA or a Conditional Access authentication context, justification, ticket number	MFA + justification
Require approval	named approvers approve each activation	for Global Admin, Owner on production
Assignment expiry	eligible or active assignments end on a date unless renewed	6-12 months
Notifications and alerts	e-mails on activation; alerts (too many Global Admins, roles assigned outside PIM)	on

- **What PIM covers** Entra roles, Azure resource roles (any scope) and PIM for Groups (membership or ownership of a group, which can carry roles)
- **Licence** Microsoft Entra ID P2 or Microsoft Entra ID Governance for the users who benefit (as of 2026)
- **Break-glass accounts** two cloud-only emergency Global Admins, permanent active, phishing-resistant MFA (FIDO2), excluded from risk-based CA, sign-ins alerted
- **Access reviews** recurring reviews of role assignments; auto-remove if not approved (SC-300 sheet for depth)

Command	Effect
<code>New-AzRoleEligibilityScheduleRequest -Name (New-Guid) -Scope \$rg -RoleDefinitionId \$owner -PrincipalId \$uid -RequestType AdminAssign -ScheduleInfoStartDateTime (Get-Date) -ExpirationType AfterDuration -ExpirationDuration P180D</code>	make a user eligible for Owner on a resource group for 180 days
<code>New-AzRoleAssignmentScheduleRequest ... -RequestType SelfActivate -ExpirationDuration PT2H -Justification 'INC-42'</code>	the user activates the eligible role for 2 hours
<code>az rest --method get --url .../roleEligibilityScheduleInstances?api-version=2020-10-01</code>	list eligible assignments (ARM REST, works where no az subcommand exists)

Workload identities

System-assigned managed identity

created with and deleted with one resource
one per resource

simplest: enable a toggle, assign roles

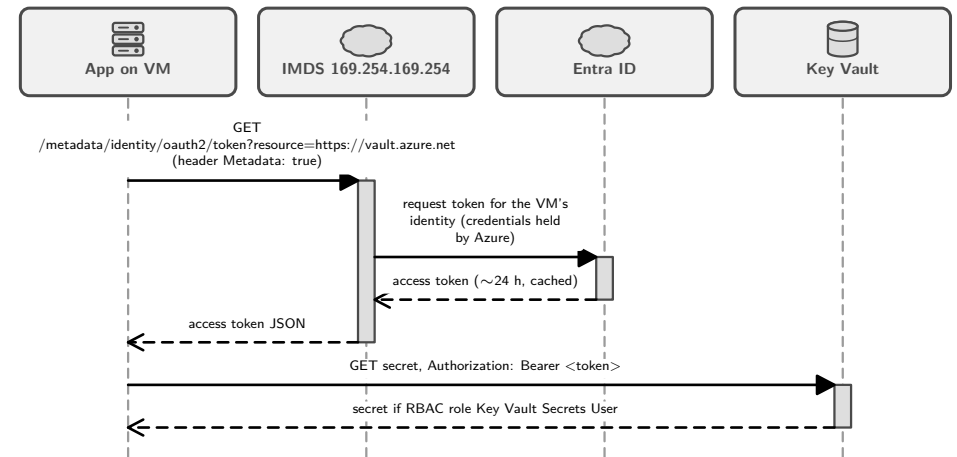
role assignments vanish when the resource is deleted

User-assigned managed identity

standalone resource with its own lifecycle
one identity shared by many resources (scale sets, many apps)

pre-create and grant roles before the resource exists (IaC, blue-green)

survives redeployment: no re-grant, no replication delay



No secret in code: the SDK's DefaultAzureCredential does steps 1-4

App credential	Use when	Risk
Managed identity	code runs on Azure (VM, App Service, Functions, AKS workload identity, Container Apps)	lowest: nothing to store or rotate
Federated credential (workload identity federation)	GitHub Actions, Azure DevOps, Kubernetes, AWS/GCP workloads	low: trust is scoped to issuer + subject (repo.org/app:environment:prod)
Certificate	workloads outside Azure without OIDC federation	medium: private key must be protected; store in Key Vault
Client secret	last resort, legacy tools	high: leaks in code and logs; max lifetime 24 months, prefer ≤ 6

☰ Delegated vs application permissions

Delegated (scope, e.g. User.Read): the app acts *as the signed-in user*; effective rights = permission $\cap$ user's rights; a user can consent to low-risk ones. **Application** (app role, e.g. User.Read.All): the app acts *as itself* with no user, across the tenant; always needs **admin consent**. Restrict user consent to verified publishers and low-impact permissions, and use the admin consent workflow.

⚠ WARNING

az ad sp create-for-rbac creates a **client secret** (no default role assignment since Azure CLI 2.37). Prefer az identity create + federated credentials for pipelines: az identity federated-credential create --issuer https://token.actions.githubusercontent.com --subject repo:org/app:environment:prod --audiences api://AzureADTokenExchange.

🔒 Authentication guardrails (recall, SC-300 goes deeper)

Mandatory MFA for Azure	Microsoft enforces MFA for the Azure portal, Entra and Intune admin centers (phase 1, from late 2024) and for Azure CLI, PowerShell, SDK and IaC write operations (phase 2, from late 2025); use workload identities for automation, never user accounts
Security defaults	free tenant-wide baseline (MFA registration, MFA for admins, block legacy auth); turn off when you build Conditional Access (Entra ID P1)
Conditional Access for admins	require phishing-resistant MFA for privileged roles; target the Windows Azure Service Management API app for Azure management sign-ins
Identity Protection	user risk and sign-in risk (P2) drive CA policies: block or require password change / MFA

Identity baseline an AZ-500 engineer must set

✓ baseline / mandatory ☐ recommended – verify in your context

- ✓ No standing Owner or Global Administrator: PIM eligible + MFA + approval
- ✓ Two break-glass accounts with FIDO2 keys, monitored sign-ins
- ✓ Role assignments to groups at the narrowest scope; no subscription Owner for apps
- ✓ Managed identities or federated credentials for every workload; no long-lived secrets
- ☐ **MUST** User consent restricted; admin consent workflow on
- ☐ **SHOULD** Quarterly access reviews of privileged roles and guest access
- ☐ **SHOULD** Alert on role assignments outside PIM (Activity log → Sentinel)

🚩 Myths, recap and self-test

YOU MAY HEAR

"Global Administrators can manage every subscription"

BUSTED

✓ ACTUALLY

Entra roles grant no Azure RBAC rights

Only after Elevate access (User Access Administrator at /) or an explicit Azure role assignment.

YOU MAY HEAR

"NotActions denies an action"

BUSTED

✓ ACTUALLY

NotActions only subtracts from that one role

Another role on the same scope can still allow it; only deny assignments block.

YOU MAY HEAR

"Owner of a storage account can read every blob with Entra ID"

BUSTED

✓ ACTUALLY

Blob data needs a data role

Storage Blob Data Reader/Contributor/Owner; Owner reads data only through account keys if shared-key access is on.

YOU MAY HEAR

"A user-assigned identity is less secure than a system-assigned one"

BUSTED

✓ ACTUALLY

Both are managed, rotated credentials

User-assigned is shareable; scope its roles tightly because every attached resource gets them.

Tick a box only when you could explain the point to someone else



Entra roles $\neq$ Azure RBAC roles

different scopes, different portals



Assignment = principal + role + scope

inherited down the hierarchy



Data actions need data roles

control-plane Owner does not read blobs or secrets



PIM for every privileged role

eligible, time-bound, MFA, approval, review



Managed identity first, federation second, certificate third, secret last

Q1 Which role can assign Azure roles but not manage resources, with limits on which roles?

A Role Based Access Control Administrator with a delegation condition (User Access Administrator has no such limit).

Q2 A Global Admin cannot see any subscription. What do they do?

A Turn on Elevate access: User Access Administrator at root /, then assign proper roles and turn it off.

Q3 Which JSON section of a role definition lists blob read/write operations?

A DataActions (control-plane operations go in Actions).

Q4 Which managed identity type to pre-grant Key Vault access before a scale set is created?

A User-assigned: it exists independently, so its role assignment can be made and replicated first.

Q5 Where does a VM get its managed identity token?

A From IMDS at http://169.254.169.254/metadata/identity/oauth2/token with header Metadata: true.

Q6 Which PIM setting forces a second person to agree before Owner is activated?

A Require approval on activation, with named approvers.

Q7 A GitHub Actions workflow must deploy to Azure without secrets. How?

A Workload identity federation: federated credential on an app or user-assigned identity, subject `repo:org/repo:environment:prod`.

Q8 Application vs delegated permission: which needs admin consent every time?

A Application permissions (the app acts as itself across the tenant).

Q9 Which lock stops deletion but allows changes?

A CanNotDelete (ReadOnly blocks changes too and can break list-keys operations).

Q10 Who can create a deny assignment?

A Only Azure itself (deployment stacks with deny settings, managed applications); not a user via `az role`.