MICROSOFT ENTRA AGENT ID

Agentic Controls and Best Practices

Securing AI Agents in the Microsoft Environment

A reference guide to identity, authentication, authorization, Zero Trust access, network controls, and governance for AI agents.

Prepared: June 2026
Source: Microsoft Entra Agent ID documentation (Microsoft Learn)
learn.microsoft.com/entra/agent-id

Contents

1. Executive Summary

Organizations are deploying AI agents at speed — assistive copilots that act for a signed-in user, autonomous agents that run on their own schedule, and digital workers that operate like team members. These agents are non-human identities that authenticate, make decisions, and take action, often against sensitive enterprise data. Treating them as ordinary applications or as human user accounts breaks down across every layer of security enforcement.

Microsoft Entra Agent ID is an identity and security framework that extends Microsoft Entra capabilities to AI agents. It provides purpose-built identity constructs and applies the same Zero Trust controls used for people and workloads — adaptive access policies, real-time risk detection, least-privilege authorization, network filtering, lifecycle governance, and full audit logging — to agent identities at enterprise scale.

This document consolidates the agentic controls available in the Microsoft environment and the operational best practices for designing, securing, and governing them. It is organized around the lifecycle of an agent identity: understanding the risk, modeling the identity, authenticating it, authorizing it, controlling its access, governing it, and monitoring it.

> **At a glance — the seven control domains**
>
> **1. Identity constructs** — blueprints, agent identities, and optional user accounts replace generic service principals.
>
> **2. Architecture planning** — decide identity type, operation pattern, blueprint count, and identities per blueprint.
>
> **3. Authentication** — OAuth 2.0 flows with managed identities or certificates instead of stored secrets.
>
> **4. Authorization** — least privilege, blocked high-risk roles and permissions, inheritable permissions.
>
> **5. Zero Trust access** — Conditional Access and ID Protection scoped to agents.
>
> **6. Network controls** — Global Secure Access filtering of agent traffic.
>
> **7. Governance & monitoring** — sponsors, access packages, reviews, and audit logs.

2. The Agentic Security Challenge

Unlike applications that execute predetermined logic, AI agents make dynamic decisions and adapt their behavior based on training data, input, and environment conditions. That adaptability is exactly what expands the organizational attack surface and is why identity-based controls are required.

2.1 Why agents are different

- **External accessibility.** Many agents interact with external users, third-party systems, or the public internet, creating pathways for adversaries to reach internal resources.

- **Permission escalation risk.** Agents are often over-provisioned for capability. An agent built to analyze financial data may be granted access to all financial records when it needs only a narrow slice.
- **Autonomous decision-making.** A compromised agent that acts autonomously can do real damage — placing unauthorized orders, modifying data, or deleting critical systems.
- **Prompt injection.** Malicious instructions hidden in data the agent processes can manipulate its behavior — a class of attack that does not affect traditional applications.
- **Agent-to-agent propagation.** A compromised orchestration agent can target the specialized agents it delegates to, propagating the compromise across a fleet.

2.2 Agent sprawl

Agent proliferation creates a governance problem known as agent sprawl — the uncontrolled expansion of agents without adequate visibility, ownership, or lifecycle control. It emerges when business units spin up agents without IT oversight (shadow AI), when temporary agents linger in production indefinitely, and when permissions exceed need and are never reviewed. The result is over-privileged agents with unclear ownership, compliance gaps when auditors expect governance over AI systems, and slow incident response because compromised agents cannot be identified quickly.

2.3 Agent security scenarios

The same agent capabilities create different risks depending on how the agent is deployed. The table below maps the four primary scenarios to their core challenge and risk.

Scenario	Description	Core challenge	Primary risk
User-initiated agent	Acts on behalf of a user, inheriting that user's access.	Prevent misuse of inherited permissions; keep the user in control and able to revoke.	Unauthorized actions performed as the user — reading files, sending messages, altering data.
Autonomous agent	Operates with its own identity and permissions, independent of any user.	Grant only what the task needs; prevent the agent exceeding its scope.	Unconstrained action — unauthorized orders, data modification, access to sensitive information.
Agent's user account	Functions as a human teammate with mailbox and collaboration access.	Keep permission scope appropriate; stop a compromised teammate spreading malware.	Access to documents and meetings, or messages sent as a trusted colleague.
Agent-to-agent (A2A)	An orchestration agent delegates to specialized agents.	Authenticate agent-to-agent communication; ensure only legitimate agents interact.	Injection of malicious agents or interception of agent interactions.

3. Agent Identity Constructs

Microsoft Entra Agent ID introduces four object types that together let agents take on governed digital identities. Understanding them is the foundation for every control that follows.

3.1 The four object types

- **Agent identity blueprint.** A template that defines the security posture — credentials, permissions, and metadata — for all agent instances of a common kind. The blueprint holds the credentials used to acquire tokens, and policies applied here are inherited by every child identity, present and future.
- **Agent identity blueprint principal.** The representation of a blueprint within a specific tenant, analogous to a service principal for a multitenant application. It lets a multitenant-capable agent create identities in a customer tenant.
- **Agent identity.** An individual instance created from a blueprint, with its own distinct entries in sign-in and audit logs. This is the right choice for most agents and the unit you disable, govern, and attribute actions to.
- **Agent's user account.** An optional account paired 1:1 with an agent identity, used only when the agent needs a user object — for example a mailbox, calendar, or Teams presence. It does not replace the agent identity; both must exist.

3.2 Why not a service principal or a user account?

Service principals are designed for deterministic, static workloads and a regular user account is designed for human sign-in. Neither fits an autonomous, adaptive agent. The comparison below shows what the agent identity adds.

Identity type	Use when…	Key characteristics
Agent identity	The agent acts on its own behalf or on behalf of users (most agents).	Enforced sponsorship, distinct audit-log entries, blueprint-managed credentials, platform restrictions on high-privilege grants.
Agent's user account	The agent needs a resource requiring a user object (mailbox, Teams channel).	Paired 1:1 with an agent identity; has a UPN, manager, and other user properties.
Service principal *(not recommended)*	A scripted, predictable workload with no autonomous decision-making.	Classic app identity without agent-specific governance, transparency, or lifecycle management.
Regular user account *(do not use)*	Never for an AI agent.	Breaks Conditional Access, distorts ID Protection's human-tuned ML, and disrupts joiner-mover-leaver governance.

Why a user account degrades security

Routing agent traffic through a human user account causes failures across every Zero Trust layer: Conditional Access controls such as MFA, compliant-device, and terms-of-use are designed for humans and fail for agents; ID Protection's machine learning is tuned for human sign-in behavior and is degraded

Identity type	Use when…	Key characteristics
for both humans and agents; and governance workflows may incorrectly remove agent access. Agents also appear in the Global Address List, Teams, and SharePoint alongside employees.		

4. Planning Your Agent Identity Architecture

Before integrating an agent with Entra Agent ID, work through four design decisions in order — earlier choices shape later ones.

4.1 Step 1 — Choose an identity type

For most agents, an agent identity is correct. Add an agent's user account only when the agent must access a system that requires a user object. Service principals and regular user accounts are not recommended for agent workloads, for the reasons in Section 3.

4.2 Step 2 — Choose an operation pattern

An agent's operation pattern determines how it acquires tokens and the context it acts in. Many agents need both patterns — for example a nightly autonomous sync plus interactive chat responses.

Characteristic	Autonomous	Interactive
User context	No user present	User is signed in
Permission type	Application permissions	Delegated permissions
Consent model	Admin consent required	User or admin consent
Token subject	Agent identity	User, with the agent as actor
Common scenarios	Background jobs, scheduled tasks, system-to-system	Chat assistants, user-facing copilots, acting on user data
Access scope	Broad, tenant-wide	Scoped to the signed-in user's data

4.3 Step 3 — Decide how many blueprints

A blueprint holds the credentials for all its child identities, so a blueprint compromise can affect every identity under it. The default is one blueprint per trust boundary — the shared risk surface where a single compromise is assumed to affect the whole perimeter. Agents sharing a runtime, secrets, file system, and network share a trust boundary and can share a blueprint; agents that cross trust domains need separate blueprints so a credential compromise cannot spread.

Not reasons to add blueprints: blocking authentication (disable a single identity or target it with Conditional Access instead), audit separation (each identity already logs separately), scale-out or replicas, and memory or context separation.

4.4 Step 4 — Decide how many identities per blueprint

The default is one agent identity per logical agent, which gives the highest-fidelity audit trail and the finest-grained access control. Use multiple identities when actions must be attributable to a specific agent, when agents have independent lifecycles, or when they have distinct responsibilities. Horizontal scale-out and shared memory pools are runtime concerns, not identity concerns, and are not reasons to create more identities.

> **Rule of thumb**
>
> **One blueprint per trust boundary; one identity per logical agent.** Start here and deviate only when a concrete security boundary or attribution requirement demands it.

5. Authentication Controls

Agents authenticate using OAuth 2.0 with specialized token-exchange patterns built on Federated Identity Credentials. Every agent is a confidential client; interactive sign-in flows are not supported for any agent object, so all authentication happens through programmatic token exchange. Microsoft strongly recommends using approved SDKs (Microsoft.Identity.Web and the Microsoft Entra ID Auth SDK sidecar) rather than implementing these flows by hand.

5.1 The three OAuth flows

Flow	When it applies	Token subject
On-behalf-of (OBO)	Interactive / assistive agents acting for a signed-in user. The agent exchanges the user's token for one scoped to the target resource.	The user, with the agent as actor
Autonomous app (client credentials)	Background, scheduled, or system-to-system work with no user present, or an interactive agent calling a backend the user cannot reach.	The agent identity
Agent's user account	Digital workers and AI teammates that operate through their own user principal (mailbox, chat).	The agent's user account

5.2 Credential management

Credential hygiene is the single most important defense against unauthorized access through an agent identity.

- **Prefer managed identities or certificates in production.** Federated identity credentials backed by managed identities eliminate stored secrets entirely and bring automatic rotation and secure storage. Certificates are the next-best option.
- **Do not use client secrets in production.** Use them only for early development or testing, and rotate them out before go-live.
- **Isolate credentials per blueprint.** Never reuse one credential across unrelated blueprints; use separate blueprints or environment-specific federated credentials for dev, test, and prod so one compromise stays contained.
- **Store private keys securely.** Keep certificate private keys in Azure Key Vault or an HSM, and limit the scope of any managed identity used for federation.
- **Rotate on a schedule.** Even though blueprints allow long-lived credentials, rotate certificates at least annually.
- **Align the flow to the scenario.** Use client credentials for autonomous agents with only the permissions they need; use OBO for interactive agents so user policies and consent apply; avoid app permissions when delegated permissions suffice.
- **Monitor token usage after deployment.** Review sign-in logs to confirm agents use the intended credential type, and periodically audit consented permissions to prevent privilege creep.

> **Boundary to remember**
> Conditional Access protects token acquisition by the agent identity or agent's user account, but it does not apply to a blueprint acquiring a token to create identities, nor to intermediate exchanges at the AAD Token Exchange Endpoint (Public), which cannot call Microsoft Graph. Agent tasks are always performed by the agent identity, not the blueprint.

6. Authorization Controls

Agent identities behave like an application or user, but with extra safeguards. Because agents act quickly and at scale, the platform deliberately limits what they can be granted so that a compromised agent cannot escalate to administrative control. Users and administrators are not permitted to consent to the most powerful permissions for an agent — even an admin cannot grant them.

6.1 Least privilege by design

Many high-privilege capabilities in Microsoft Entra assume a careful human administrator. An unrestrained agent holding those privileges could perform far-reaching actions such as deleting users or changing security settings. The platform therefore blocks the most sensitive roles and permissions outright and expects every other grant to follow least privilege. The list of allowed roles and permissions evolves over time.

6.2 Microsoft Entra role assignments

An agent identity can be assigned certain built-in Entra roles, but highly privileged directory roles are blocked. Custom roles cannot be assigned to agents, and agents cannot be members of role-assignable groups. Microsoft also provides the Agent ID Administrator and Agent ID Developer roles for managing and creating agents themselves.

Blocked for agents (examples)	Allowed for agents (examples)
Global Administrator	Global Reader, Security Reader, Directory Readers
Privileged Role Administrator	Reports Reader, Usage Summary Reports Reader
User Administrator	Exchange / SharePoint / Teams Administrator, AI Administrator
Any custom role; membership in role-assignable groups	Compliance Administrator, License Administrator, and other lower-privilege roles

Note. *The allowed-role list is extensive and managed by Microsoft; the rows above are representative, not exhaustive. Always select the least-privileged role that meets the need.*

6.3 Microsoft Graph permissions

Agents use the same Microsoft Graph permission model as other apps and may request delegated or application permissions — except for a set of high-risk permissions that are blocked entirely.

Blocked Graph permission	Why it is blocked
`Application.ReadWrite.All`	Allows managing all applications.
`RoleManagement.ReadWrite.All`	Full control over users, groups, roles, and directory settings.
`User.ReadWrite.All`	Full control of all user accounts.
`Directory.AccessAsUser.All`	Directory access as the signed-in user; cannot be consented even by an admin.

Lower-privilege, bounded permissions remain available. An agent that needs to read a user's mail or files on that user's behalf can request a delegated scope such as Mail.Read or Files.Read and have the user or admin consent. What is blocked is the tenant-scoped, cross-user, or administrative control — agents operate under a principle of limited scope, able to do only what a regular user could consent to or what an admin explicitly grants in a controlled, scoped manner.

6.4 Choosing the right grant

The agent needs to…	Use	Notes
Access specific Azure resources	Azure roles	Scope narrowly to the resource or resource group (e.g., Key Vault Reader on one vault).
Perform directory-level actions	Microsoft Entra roles	Only if a suitable lower-privilege role exists; otherwise rely on Graph permissions.
Act on behalf of a user	Delegated Graph permissions	Requires user consent; the agent cannot exceed that user's access.
Run autonomously across the tenant	Application Graph permissions	Use sparingly; grant only specific, non-high-privilege permissions with explicit admin consent.

6.5 Inheritable permissions

Blueprints support inheritable permissions, which let an administrator grant a permission once at the blueprint level and have it flow automatically to all present and future agent identities created from that blueprint — eliminating repeated consent across environments and business units. Two configurations work together:

- **Required resource access.** The blueprint's up-front declaration of the APIs and permissions its child identities need. It makes the consent decision explicit and reviewable for administrators but does not itself grant anything.
- **Inheritable permissions.** The list of resource apps whose permissions are eligible to flow to child identities. For inheritance to occur, the resource must be listed here and the permission must be granted (via static consent in required resource access, or via dynamic consent with the permission explicitly requested).

Visibility caveat. Inherited permissions are not shown on agent identities in the admin center or Microsoft Graph; they are only observable in the token at runtime, where the platform merges inherited and directly granted permissions.

Best practice for inheritable permissions is to minimize up-front grants, predeclare resource apps that might be needed later so admins can anticipate them, make a required permission's resource app inheritable so it need not be granted on every identity, and reserve non-inheritable, per-identity consent for highly privileged operations that warrant a fresh decision each time.

7. Zero Trust Access Controls

Microsoft Entra applies the same Zero Trust controls to agents as to people, through Conditional Access and Identity Protection. Both should be configured specifically for agents rather than relying on user-targeted policies.

7.1 Conditional Access for agents

Conditional Access evaluates the subject requesting access and the audience (resource) being accessed whenever Entra issues or refreshes a token. Each access token has exactly one subject and one audience, so an agent accessing several resources typically needs a separate token — and a separate policy evaluation — for each. The control point depends on the access pattern:

- **Acting on behalf of a user (OBO).** The user is the subject, so policies target users and groups; the OBO token exchange is itself evaluated, letting admins control which resources an agent may reach on the user's behalf.
- **Acting as an application.** The token is issued to the agent identity, so policies are scoped to the agent identity.
- **Acting as a user account.** The token is issued to the agent's user account, and policy is evaluated against that account.

Scaling Conditional Access

Two mechanisms make agent policy manageable at scale. Applying a policy at the blueprint level automatically covers every identity derived from that blueprint, including future ones (it does not cover agents' user accounts). Custom security attributes — business labels such as Environment, Department, or DataSensitivity — let you target whole classes of agents and resources, with policies applying automatically to any agent that matches.

> **Conditional Access limits to plan around**
>
> **Agents cannot satisfy interactive controls.** MFA, compliant-device, and terms-of-use controls are designed for humans; create dedicated agent policies using identity filters, risk signals, and named locations instead.
>
> **Coverage gaps.** Policies targeting “all users” do not include agents' user accounts; a policy targeting agent identities does not apply to the agent's user account; and policies cannot scope agents' user accounts by group membership.
>
> **Out of scope entirely.** Conditional Access protects only Entra-secured resources. An agent calling a resource with an API key bypasses the Entra token pipeline. Security defaults also bypass Conditional Access.

Tip. Use report-only mode to test agent policies before enforcing them, and audit broad policies (such as “all users must use MFA”) so they do not unintentionally block agent flows.

7.2 Identity Protection for agents

Microsoft Entra ID Protection detects identity-based risk on agents and feeds risk signals to Conditional Access. A Learning Mode suppresses behavioral alerts for agents that lack activity history — preventing false positives during onboarding and after inactivity — while a parallel detection still catches genuinely malicious early-life behavior. In OBO flows, risky activity is attributed to the user, not the agent, so remediation targets the compromised session without disrupting the agent for everyone else.

Risk detections that can flag an autonomous agent include:

Detection	What it indicates
Confirmed compromised	An administrator confirmed the agent is compromised.
Early-life malicious activity	A newly created agent immediately exhibited multiple attacker-like patterns.
Entra directory reconnaissance	Suspicious reconnaissance or high-risk directory operations.
Failed access attempt	Attempts to reach unauthorized resources — possible token replay.
Microsoft Entra threat intelligence	Activity matching known attack patterns from Microsoft threat intelligence.
Sign-in spike	A sign-in volume far above the agent's norm — possible automation or toolkit.
Suspicious credential usage	New credentials added to a blueprint and then actually used.
Unfamiliar resource access	The agent targeted resources it does not normally access.

From the Risky Agents report, administrators can confirm compromise (which sets risk to High and triggers risk-based blocking policies), confirm safe, dismiss risk, or disable the agent across Entra and connected apps. Risk data is queryable through the riskyAgents and agentRiskDetections Graph collections, retained for up to 90 days, and exportable to Log Analytics, storage, an event hub, or a SIEM.

Recommended baseline. Deploy a Conditional Access policy that blocks agents flagged High risk — analogous to blocking risky users — so a compromised agent is cut off immediately. Microsoft Managed Policies and a ready-made template are available to simplify this.

8. Network Controls

Global Secure Access extends the network security policies you already apply to users to agent traffic, giving administrators visibility and control over how agents reach external resources. For Microsoft Copilot Studio agents, you enable traffic forwarding in the Power Platform Admin Center on a per-environment or per-environment-group basis; forwarded traffic includes HTTP node traffic, custom connectors, and the MCP server connector.

Once agent traffic flows through the Global Secure Access proxy, it is evaluated against your security policies just like user traffic. Policies are configured through the tenant-level baseline profile for consistent enforcement across all agent traffic. Available controls include:

- **Web content filtering and web categorization** to control which APIs and MCP servers agents can reach.
- **Threat-intelligence filtering** to automatically block and alert on known-malicious destinations.
- **Network file filtering** to restrict uploads and downloads by file type and minimize data-movement risk.
- **Prompt-injection detection** to identify and block attempts to manipulate agent behavior through malicious instructions.
- **Network activity logging** to remote tools for audit and threat detection.

9. Governance and Lifecycle

Microsoft Entra ID Governance extends lifecycle management, entitlement management, and access reviews to agent identities so that access is intentional, auditable, time-bound, and always backed by an accountable human.

9.1 The administrative model: owners, sponsors, managers

Agent ID separates technical administration from business accountability so that oversight does not require excessive permissions.

Role	Responsibility	Authority
Owner	Technical administrator — setup, configuration, credential management.	Edit settings and authentication properties, manage credentials, add owners and sponsors, disable, re-enable, restore, or hard-delete identities. Optional. Users or service principals (not groups).
Sponsor	Business accountability for the agent's purpose and lifecycle decisions.	Non-destructive lifecycle only: enable/disable, modify sponsors, soft-delete, request access packages, give approvals. Required on every blueprint and agent identity. Users or select groups.
Manager	Organizational-hierarchy owner of the agent's user account.	Request access packages for reporting agents; cannot modify or delete. Optional.

Succession is built in. If a sponsor leaves the organization, sponsorship transfers automatically to their manager, so a human is always accountable. Lifecycle Workflows can notify cosponsors and managers of impending sponsorship changes. A sponsor is required at creation for agent identities and blueprints; for app-only creation the creating service must set a sponsor, and for delegated creation the calling user becomes sponsor only if none is specified.

9.2 Access packages and entitlement management

Beyond the limited permissions inherited from their blueprint, agent identities receive resource access through access packages, which grant time-bound, auditable access via approval workflows rather than direct assignment. Access packages can grant security-group memberships, application OAuth/Graph permissions, and (lower-privilege) Microsoft Entra roles. To enable them, configure an assignment policy whose “Who can get access” setting includes users, service principals, and agent identities, then select All agents.

Agents can be assigned access through three pathways: the agent identity can programmatically request a package when it needs one; the sponsor can request on the agent's behalf, providing human oversight; or an administrator can assign the identity directly. When an assignment has an expiry date and a sponsor is set, the sponsor is notified as expiry approaches and can request an extension (triggering a fresh approval cycle) or let it lapse — at which point access is removed automatically.

9.3 Access reviews and combating sprawl

- **Include agents in access reviews** every 6–12 months, with sponsors attesting that each agent is still needed and correctly configured; decommission those that are not confirmed.
- **Run a quarterly orphan review** to find agents with missing sponsors, stale metadata, or no recent activity, and reassign or retire them.
- **Register every agent in Entra** through the Agent ID framework — whether built in Copilot Studio, Foundry, Azure, Teams, or an external platform — to eliminate shadow AI and give IT a complete inventory through the admin center and Microsoft Graph.
- **Standardize naming** (for example a department or function prefix such as Agent-HROnboardingBot) so agents are recognizable in logs and the admin center.

10. Monitoring and Audit

All agent authentication and activity is logged in Microsoft Entra ID for compliance and audit. Continuous monitoring keeps agents within expected boundaries.

- **Watch sign-in logs for anomalies** — spikes in token requests, access to unexpected APIs, or sign-ins from unfamiliar IP ranges. Each entry shows the resource, credential type, and outcome.
- **Track configuration changes in audit logs** — blueprint changes, credential additions, permission grants, and role assignments — and alert on anything outside your normal deployment pipeline.
- **Set proactive alerts** for credential expiration, agents blocked by Conditional Access or ID Protection, excessive failed token acquisitions, and unexpected permission or role changes.
- **Fold agents into incident response** — when investigating an incident, check whether any agent had access to affected resources and review its activity during the window.
- **Retain logs for compliance** for the duration your framework requires, exporting high-volume agent logs to a secure archive via diagnostic settings where needed.

11. Consolidated Best-Practices Checklist

Identity design

- ☐ Plan blueprints before deploying agents; define settings, permissions, and metadata up front.
- ☐ Provision a unique identity per agent instance; never share identities between agents.
- ☐ Assign a sponsor and an owner at creation and verify them when personnel change.
- ☐ Fill in description, tags, and verified-publisher metadata on every blueprint.
- ☐ Apply policies, permissions, and governance at the blueprint level for inheritance and a kill-switch.
- ☐ Use the Agent ID framework for all agents — never plain app registrations or service principals.
- ☐ Create agents' user accounts only when a user object is genuinely required.

Credentials

- ☐ Use managed identities (federated credentials) or certificates in production; avoid client secrets.
- ☐ Isolate credentials per blueprint and per environment.
- ☐ Store private keys in Key Vault or an HSM; rotate certificates at least annually.
- ☐ Match the OAuth flow to the scenario; prefer delegated over application permissions.

Access control

- ☐ Segment agents with custom security attributes and target them in Conditional Access.
- ☐ Block High-risk agents automatically with a risk-based Conditional Access policy.
- ☐ Create agent-specific Conditional Access policies; test in report-only mode first.
- ☐ Audit broad user policies so they do not unintentionally block — or fail to cover — agents.
- ☐ Grant least privilege; right-size permissions to specific scopes, resources, or sites and review periodically.

Governance

- ☐ Register every agent centrally to eliminate shadow AI.
- ☐ Standardize naming conventions across agents.
- ☐ Include agents in access reviews every 6–12 months with sponsor attestation.
- ☐ Run quarterly reviews for orphaned or inactive agents.
- ☐ Use access packages for time-bound, approval-based access instead of direct grants.

Monitoring

- ☐ Alert on sign-in anomalies and out-of-pipeline configuration changes.
- ☐ Set proactive alerts for credential expiry, risk blocks, and permission changes.

- ☐ Include agent identities in incident response and post-mortems.
- ☐ Retain agent logs per your compliance framework.

Dev / IT alignment

- ☐ Build through supported creation channels (Copilot Studio, Graph APIs, Agent 365 CLI).
- ☐ Run a production handshake: an identity admin verifies blueprint, sponsor, permissions, and policies.
- ☐ Validate auth flows and policies in a sandbox or separate dev tenant first.
- ☐ Treat agent configurations as code in source control to prevent drift.

12. Licensing Summary

Agent ID itself — creating and managing agent identities and blueprints — is available to all Microsoft Entra customers. Operating agents across Microsoft 365 (via Microsoft Agent 365) requires a Microsoft Agent 365 license per user. Extending Entra security features to agents requires Microsoft 365 E7 (which includes Agent 365 and the Microsoft Entra Suite) or Microsoft 365 E5 paired with a Microsoft Agent 365 license. Customers without E5 or E7 can license individual capabilities standalone alongside Agent 365:

Capability	Standalone license requirement
Conditional Access for agents	Microsoft Entra ID P1
Identity Protection for agents	Microsoft Entra ID P2
Identity Governance for agents	Microsoft Entra ID P1
Network controls for agents	Microsoft Entra Internet Access (in the Entra Suite or licensed separately)

Licensing for agent capabilities is evolving and enforcement timelines change. Verify current requirements and pricing against Microsoft's official documentation before planning a rollout.

Appendix — Source Documentation

This guide synthesizes the Microsoft Entra Agent ID documentation set on Microsoft Learn. Key pages:

- What is Microsoft Entra Agent ID? — https://learn.microsoft.com/en-us/entra/agent-id/what-is-microsoft-entra-agent-id
- Microsoft Entra security for AI overview — https://learn.microsoft.com/en-us/entra/agent-id/security-for-ai-overview
- Plan your agent identity architecture — https://learn.microsoft.com/en-us/entra/agent-id/how-to-plan-agent-identity-architecture
- Best practices for Microsoft Entra Agent ID — https://learn.microsoft.com/en-us/entra/agent-id/best-practices-agent-id
- Authentication protocols in agents — https://learn.microsoft.com/en-us/entra/agent-id/agent-oauth-protocols
- Authorization in Microsoft Entra Agent ID — https://learn.microsoft.com/en-us/entra/agent-id/authorization-agent-id
- Inheritable permissions and required resource access — https://learn.microsoft.com/en-us/entra/agent-id/concept-inheritable-permissions
- Administrative relationships (owners, sponsors, managers) — https://learn.microsoft.com/en-us/entra/agent-id/agent-owners-sponsors-managers
- Conditional Access for agents — https://learn.microsoft.com/en-us/entra/identity/conditional-access/agent-id
- ID Protection for agents — https://learn.microsoft.com/en-us/entra/id-protection/concept-risky-agents
- Governing agent identities — https://learn.microsoft.com/en-us/entra/id-governance/agent-id-governance-overview
- Secure Web and AI Gateway for agents — https://learn.microsoft.com/en-us/entra/global-secure-access/concept-secure-web-ai-gateway-agents

Disclaimer. *This document is a synthesized reference for internal use. Microsoft's product capabilities, blocked/allowed lists, and licensing change over time; always confirm specifics against the official Microsoft Learn documentation before implementation.*