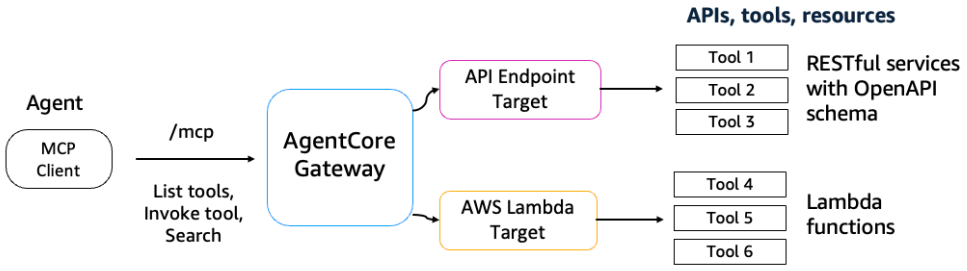
AgentCore Gateway

Amazon Bedrock AgentCore Gateway

What is AgentCore Gateway?

AI agents need tools to perform real-world tasks — from querying databases to sending messages to analyzing documents. With Amazon Bedrock AgentCore Gateway, developers can convert APIs, Lambda functions, and existing services into MCP-compatible tools and make them available to agents through Gateway endpoints with just a few lines of code. Gateway supports OpenAPI, Smithy, and AWS Lambda as input types, and is the only solution that provides both comprehensive ingress authentication and egress authentication in a fully-managed service. AgentCore Gateway also provides 1-click integration with several popular tools such as Salesforce, Slack, Jira, Asana, and Zendesk.

Learn more about AgentCore Gateway



Key Components

- **Gateway Instance** – The MCP server URL that an agent connects to. Acts as the single ingress point and enforces inbound OAuth.
- **Gateway Targets** – Declarative bindings that map tool definitions to back-end resources. Supported types:
  - **OpenAPI specification** – REST services described in OpenAPI 3.0/3.1.
  - **Lambda function** – Business logic packaged as AWS Lambda.
  - **Smithy model** – Interface definitions that generate strongly typed tools.

Learn more about AgentCore Gateway's key components

- **Credential Provider Configuration** – Outbound authentication contract for each target. Modes:
  - **GATEWAY_IAM_ROLE** – Calls AWS services with the Gateway execution role.
  - **API_KEY** – Injects a header or query-string key retrieved from the Token Vault.
  - **OAUTH** – Performs two-legged OAuth client-credentials flow, caches tokens, refreshes them pre-expiry.
- **Authorizer** – The OAuth/OIDC configuration that verifies caller tokens before any MCP request is processed.
- **Security Guard** – Service component that binds inbound identities to outbound scopes and ensures policy compliance.

- **Translation Engine** – Converts MCP JSON-RPC into REST or Lambda payloads and normalises responses back to MCP result envelopes.
- **Semantic Search Index** – Catalogues every tool with embeddings so agents can issue `x-amz-bedrock-agentcore-search` queries instead of receiving large static manifests.
- **Observability Pipeline** – Emits per-call latency, status, error taxonomy, and target-level invocation counts to CloudWatch; integrates with OTEL for trace correlation.

How it works (General Flow)

1. **Gateway Creation** – Builder calls `CreateMcpGateway`, supplying an OAuth authorizer configuration and an execution IAM role. The service returns an MCP endpoint URL. Learn more about setting up an AgentCore Gateway
2. **Target Registration** – Using `CreateGatewayTarget`, the builder attaches Lambda, OpenAPI, or Smithy definitions. Each target specifies its credential provider so AgentCore Gateway knows how to sign the outbound request. Learn more about target registration in AgentCore Gateway
3. **Agent Invocation**
   - The agent obtains an access token from the same OIDC provider, then calls the MCP endpoint with `tools/list` or `tools/call`.
   - AgentCore Gateway validates the token, resolves the requested tool to a target, and inspects the credential provider.
   - For AWS targets, it signs the request with SigV4 using its execution role; for third-party APIs it injects the stored API key or fetches an OAuth access token.
   - Translation Engine rewrites the payload to match the target protocol, forwards the call, receives the HTTP or Lambda response, converts it back to an MCP result, and streams it to the agent.
4. **Semantic Tool Selection** – When the agent sends `tools/call` to the built-in search tool with a natural-language query, AgentCore Gateway queries its embedding index and returns a ranked list

Event ends in 17 hours 53 minutes.

Labs

In the following labs, you'll learn how to:

- **Lab: MCPify your AWS Lambda** - convert existing AWS Lambda functions into fully managed MCP servers without managing infrastructure.
- **Lab: Transform OpenAPI APIs into MCP Tools** - transform existing REST APIs defined in OpenAPI specifications into MCP tools with authentication.
- **Lab: Transform Smithy APIs into MCP Tools** - transform existing Smithy API specifications into MCP tools, enabling operational AI agents to interact with AWS services like S3 and DynamoDB.
- **Lab: Gateway Search Tools** - implement semantic search capabilities to efficiently manage and discover tools in large enterprise tool sets.

Previous Next