GUIDE

AWS MCP Server Now Generally Available

> ⚠️ **HIGH PRIORITY** — Important fixes. Upgrade soon.

> **Version:** GA | **Released:** 2026-05-19 | **Upgrade from:** Preview

---

Release at a Glance

Today marks a significant milestone for AI-native development on AWS: the **AWS Model Context Protocol (MCP) Server is now Generally Available (GA)**. This release moves the server out of its preview phase, providing a robust, managed solution for AI agents and AI-native IDEs to interact securely and audibly with AWS services.

Here's the TL;DR for developers:

- **Managed Access to AWS APIs:** The AWS MCP Server provides a managed remote server, enabling AI coding agents and AI-native IDEs to access over 15,000 AWS APIs.
- **Empower AI Agents:** Facilitates foundation models in performing real-world, multi-step tasks across AWS services through natural language interactions.
- **Migration Required:** If you were using `awslabs.core-mcp-server` or certain deprecated MCP servers, you **must** migrate to the new AWS MCP Server or configure individual servers directly in your clients.
- **Immediate Upgrade:** This is a high-urgency upgrade for all developers building AI agents or AI-native IDEs that interact with AWS, especially those on preview versions.

Headline New Features

The GA release of the AWS MCP Server brings a suite of powerful capabilities designed to bridge the gap between AI models and the vast AWS ecosystem. These features are not just incremental improvements but fundamental enablers for a new generation of AI-driven development.

Managed Access to Over 15,000 AWS APIs

The core offering is a **managed remote Model Context Protocol (MCP) server** for AWS services. This means developers no longer need to self-host or manage the infrastructure for AI agents to interact with AWS. The server provides a secure conduit to **over 15,000 AWS APIs**, allowing foundation models to perform actions across the entire AWS landscape.

Secure and Auditable AI-Native Interactions

A critical concern for AI agents operating within production environments is security and compliance. The AWS MCP Server is engineered to enable AI coding agents and AI-native IDEs to **securely and audibly access AWS APIs**. This includes robust authentication, authorization, and logging capabilities, ensuring that AI-driven actions are traceable and adhere to organizational security policies.

Natural Language Interaction with AWS Services

This release transforms how foundation models interact with AWS. By leveraging the MCP, models can now **interact with AWS APIs through natural language**. This capability moves beyond simple API calls, allowing AI agents to understand and execute complex, multi-step tasks described in human-like language, significantly reducing the cognitive load for developers.

Authoritative AWS Knowledge Integration

Beyond API execution, the server provides **access to authoritative AWS knowledge for AI agents and MCP clients**. This means AI agents can not only perform actions but also reason about AWS best practices, service configurations, and troubleshooting, drawing directly from official AWS documentation and guidelines. This feature enhances the intelligence and reliability of AI-driven operations.

Out-of-the-Box Client Compatibility

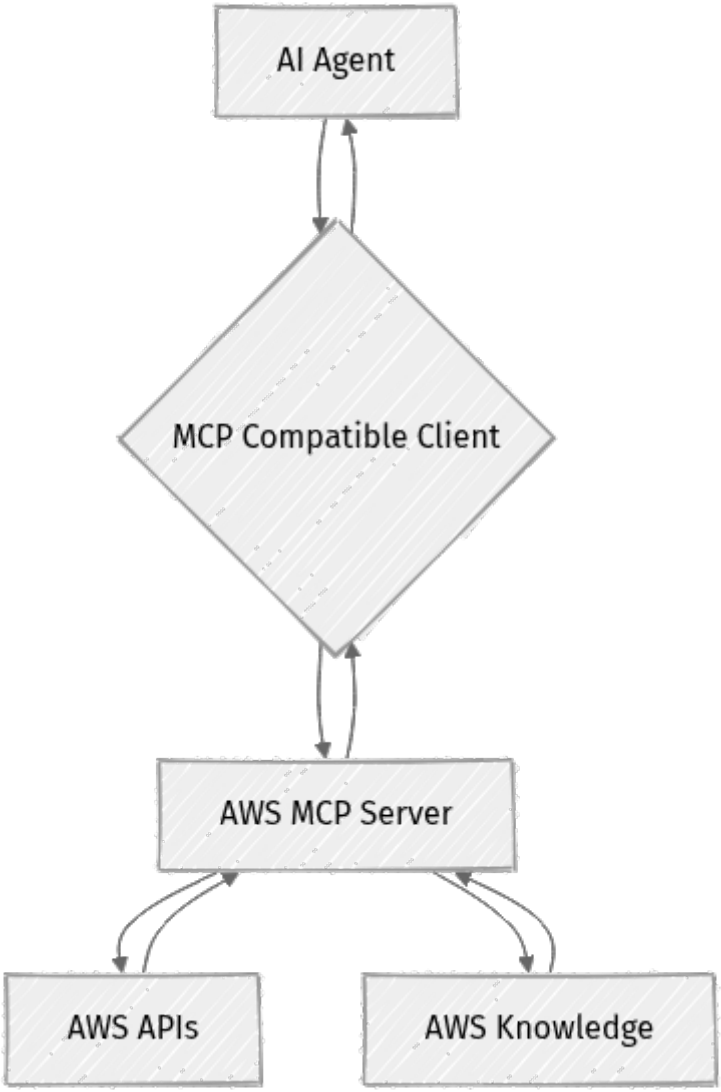
To accelerate adoption, the AWS MCP Server offers **out-of-the-box compatibility with leading MCP-compatible clients** such as Claude Code, Kiro, and Cursor. This ensures that developers using these popular AI coding assistants and IDEs can seamlessly integrate with the new AWS MCP Server without extensive configuration or custom development.

How it Integrates with the Model Context Protocol

The Model Context Protocol (MCP) is designed to standardize how AI agents and tools interact with external systems and knowledge bases. The AWS MCP Server acts as a crucial implementation of this protocol, specifically tailored for the AWS ecosystem.

📌 **Key Idea:** The AWS MCP Server is a managed MCP endpoint that translates AI agent requests into AWS API calls and retrieves AWS knowledge, all while enforcing security and auditability.

Here's a simplified flow:



1. **AI Agent / IDE Initiates Request:** An AI agent or an AI-native IDE (e.g., Claude Code, Kiro, Cursor) formulates a task or query, often in natural language.
2. **MCP Client Translation:** An MCP-compatible client translates this request into a structured MCP message.

3. **AWS MCP Server Processing:** The AWS MCP Server receives the MCP message. It then:
    - **Authenticates and Authorizes:** Verifies the identity and permissions of the requesting client/agent.
    - **Translates to AWS API Calls:** Interprets the request and translates it into appropriate AWS API calls (e.g., `ec2:RunInstances`, `s3:PutObject`).
    - **Accesses AWS Knowledge:** If the request requires information, it queries authoritative AWS knowledge bases.
    - **Executes and Monitors:** Executes the AWS API calls and monitors their outcome.
    - **Audits:** Logs all interactions for compliance and debugging.
4. **Response Back to Client:** The server formats the results (API responses, knowledge snippets, execution status) back into an MCP message.
5. **Client and Agent Interpretation:** The MCP client receives and translates the response for the AI agent or IDE, which can then present the information or take further actions.

This integration provides a standardized, secure, and scalable way for AI to operate within the AWS cloud, abstracting away the complexities of direct AWS API interaction.

Target Use Cases

The AWS MCP Server unlocks a new paradigm for interacting with AWS, moving beyond traditional scripting and manual console operations. It's designed for developers and teams building the next generation of intelligent cloud solutions.

- **AI Coding Agents:** Empower AI agents to write, debug, and deploy infrastructure as code, manage deployments, and perform operational tasks across AWS services using natural language prompts.
    - Example: An agent could respond to "Deploy a new web application with a load balancer and auto-scaling group" by provisioning the necessary EC2 instances, ALB, and ASG configurations.

- **AI-Native IDEs:** Integrate deep AWS context and automation directly into developer environments, allowing developers to interact with AWS resources through conversational interfaces or intelligent code completion.
    - Example: A developer types a comment like "create an S3 bucket for logs," and the IDE's AI assistant generates the `aws s3 mb` command or the CloudFormation/Terraform snippet.
- **Foundation Models for Cloud Operations:** Enable large language models (LLMs) to perform complex, multi-step operational tasks, such as diagnosing issues, applying patches, or optimizing resource usage based on real-time metrics and policy.
    - Example: An LLM could analyze CloudWatch alarms, identify a misconfigured security group, and then use the MCP server to update the security group rules.
- **Automated Cloud Governance and Compliance:** Develop AI agents that continuously monitor AWS environments for compliance violations, security misconfigurations, or cost inefficiencies, and automatically suggest or apply corrective actions.
    - Example: An agent detects an S3 bucket without encryption, accesses AWS knowledge on encryption best practices, and then uses the MCP server to enable server-side encryption.
- **Intelligent Troubleshooting and Support:** Build AI-powered chatbots or virtual assistants that can understand user problems related to AWS services, query AWS APIs for diagnostic information, and provide step-by-step resolution guides.
    - Example: A user asks, "Why is my Lambda function failing?" The agent checks CloudWatch logs, Lambda configurations, and IAM permissions via the MCP server to pinpoint the issue.

Breaking Changes and Migration Path

This GA release introduces important breaking changes, particularly for users of the previous `awslabs.core-mcp-server` and other deprecated MCP servers. A high-priority migration is required to leverage the new managed AWS MCP Server.

Migration from `awslabs.core-mcp-server`

What Changed: The `awslabs.core-mcp-server` package, previously used for running a core MCP server, is no longer the recommended approach. The new AWS MCP Server is a managed service, abstracting away the need to deploy and manage a generic core server yourself.

Impact: Clients configured to use `awslabs.core-mcp-server` will cease to function correctly or will not be able to access the latest features and managed capabilities of the AWS MCP Server.

Migration Path: Users must migrate from using `awslabs.core-mcp-server` to **configuring individual MCP servers directly in their clients**. This means your MCP client (e.g., Claude Code, Kiro, Cursor) will now directly connect to the managed AWS MCP Server endpoint rather than an instance of `awslabs.core-mcp-server`.

Refer to the official migration guide for detailed instructions: mcp/docs/migration-core.md

Deprecation of Certain MCP Servers (as of March 2026)

What Changed: Certain previously available MCP servers were deprecated as of March 2026. While the specific list of deprecated servers is extensive, the general principle is that these older, often community-contributed or specialized, servers are being consolidated or replaced by the official, managed AWS MCP Server.

Impact: Applications and AI agents relying on these deprecated servers will eventually lose functionality or access to updates. Continued use is not recommended and poses a risk of service interruption.

Migration Path: Developers using any of these deprecated MCP servers are required to **migrate to the new AWS MCP Server** for a managed, secure, and feature-rich experience. Alternatively, for highly specific use cases not covered by the managed service, you might need to configure specific individual server implementations directly within your client, ensuring they are compatible with the latest MCP specification.

Consult the official AWS documentation and the migration guide for a complete list of deprecated servers and their recommended migration paths.

What Developers Need to Know to Get Started

Getting started with the AWS MCP Server involves provisioning the service, configuring your AI agents or IDEs, and understanding the security implications.

1. **Provisioning the AWS MCP Server:**
    - The AWS MCP Server is a managed service. You can provision it through the AWS Management Console, AWS CLI, or AWS SDKs.
    - Look for the service under the AI/ML category or search for "MCP Server" in the console.
    - During provisioning, you will define access policies and permissions for your AI agents, ensuring they only have the necessary scope to interact with AWS APIs.
2. **Configuring Your MCP-Compatible Clients:**
    - **AI-Native IDEs (e.g., Claude Code, Kiro, Cursor):** These clients typically have configuration settings to specify an MCP server endpoint. You will need to provide the endpoint URL of your newly provisioned AWS MCP Server.

```
// Example (conceptual, actual configuration varies by client)
{
  "mcpServer": {
    "endpoint": "https://your-aws-mcp-server-id.mcp.aws",
    "region": "us-east-1",
    "credentials": {
      "type": "aws_iam_role",
      "roleArn": "arn:aws:iam::123456789012:role/AI_Agent_Role"
    }
  }
}
```

```
- **Custom AI Agents / Foundation Models:** If you're building a custom agent, your code will need to instantiate an MCP client library (if available for your language) and configure it to point to the AWS MCP Server endpoint.
```

```
Ensure your agent's execution environment has the necessary AWS IAM
permissions to interact with the AWS MCP Server.
```

1. **IAM Permissions and Security:**
    - **Least Privilege:** Always adhere to the principle of least privilege. Grant your AI agents only the IAM permissions required to perform their specific tasks via the AWS MCP Server.
    - **Auditability:** Leverage AWS CloudTrail and CloudWatch logs to monitor and audit all interactions between your AI agents and AWS services facilitated by the MCP Server. This is crucial for security and compliance.
2. **Experimentation:**
    - Start with simple tasks to confirm connectivity and basic functionality.
    - Gradually increase complexity, testing multi-step workflows and natural language interactions.

For a comprehensive guide on initial setup and configuration, refer to the official AWS MCP Server User Guide.

Official Resources

- **Official Changelog / What's New Announcement:** https://aws.amazon.com/about-aws/whats-new/2026/05/aws-mcp-server/
- **Migration Guide from Core MCP Server:** https://github.com/awslabs/mcp/blob/main/docs/migration-core.md
- **AWS MCP Server User Guide:** (Search evidence indicates this exists, but no direct URL provided in the brief. It would typically be linked from the main GA announcement.)