User Guide

AWS FinOps Agent (preview)

AWS FinOps Agent (preview): User Guide

Table of Contents

..... vi
What is AWS FinOps Agent (preview)? 1
Related services 1
Getting started with AWS FinOps Agent 3
Creating an agent 3
Prerequisites 3
The creation wizard 4
Verifying your agent setup 6
Accessing the web application 6
Sharing access with your team 6
Managing your agent after creation 6
Next steps 7
Using the web application 7
Accessing the web application 7
Layout overview 8
Side navigation 8
Chat 9
Workspaces 9
Approvals 10
Providing feedback 10
IAM setup guide 11
Step 1: Create IAM policies 12
Step 2: Create two IAM roles 20
Activate dependent AWS services 21
Working with AWS FinOps Agent 23
Use cases 23
Asking about costs 24
Investigating cost anomalies 25
Event-triggered cost anomaly investigation 26
Optimization recommendations 28
Generating cost reports 30
Task management 31
On-demand tasks 31
Scheduled tasks 32

Event-based tasks ... 32
Approvals for read-write actions ... 32
Context files and memory ... 33
Apply your accounts, owners, and policies with context files ... 33
Supported file types and limits ... 34
Manage context files ... 35
Agent memory across sessions ... 35
Long conversation handling ... 36
Enable Jira with AWS FinOps Agent ... 36
Integration and connection model ... 36
Prerequisites ... 36
Register a Jira integration ... 38
Create a Jira connection ... 38
Capabilities ... 39
Issue details ... 39
Integration statuses ... 39
Approval workflow ... 40
Error handling ... 40
Enable Slack with AWS FinOps Agent ... 41
Integration and connection model ... 41
Prerequisites ... 41
Register a Slack integration ... 41
Create a Slack connection ... 42
Capabilities ... 43
Message details ... 43
Integration statuses ... 43
Error handling ... 44
Monitoring and observability ... 45
CloudTrail logs ... 45
AWS FinOps Agent information in CloudTrail ... 45
How caller identity appears in CloudTrail ... 46
What is not logged in CloudTrail ... 46
Security ... 47
Data protection ... 47
Data storage and encryption ... 48
Data isolation across agents ... 49

Managing your data ... 49
Identity and access management ... 50
Audience ... 50
Authenticating with identities ... 51
Managing access using policies ... 52
How AWS FinOps Agent works with IAM ... 54
Identity-based policy examples ... 59
Troubleshooting ... 62
Agent guardrail controls ... 63
Read-only actions ... 63
Read-write actions ... 64
Behavioral guardrails ... 64
Generative AI accuracy considerations ... 64
Service improvement ... 65
Content used for service improvement ... 65
How to opt out ... 65
Cross-region inference ... 65
Quotas ... 67
Requesting a quota increase ... 68
Document history ... 69

AWS FinOps Agent is in preview release and is subject to change.

What is AWS FinOps Agent (preview)?

> **Note**
>
> **Powered by Amazon Bedrock**: AWS FinOps Agent is built on Amazon Bedrock and includes automated abuse detection implemented in Amazon Bedrock to enforce safety, security, and the responsible use of AI.

AWS FinOps Agent is a frontier agent that makes it easy for customers to continuously monitor costs, investigate anomalies, and surface optimization opportunities across their cloud environments.

AWS FinOps Agent provides:

- **Event-triggered cost anomaly investigation.** Configure the agent to listen for AWS Cost Anomaly Detection events and produce a consolidated investigation report, delivered to Jira or Slack.
- **Cost inquiry in natural language.** Ask cost questions about your workloads and get answers using your cost and usage data.
- **Recurring cost reporting.** Schedule recurring cost reports (for example, daily, weekly, or monthly) rendered in a downloadable, presentation-ready format in HTML, PDF, or PPT.
- **Optimization opportunities in one place.** Pull recommendations from AWS Cost Optimization Hub and AWS Compute Optimizer, and summarize them into a Jira ticket so the engineering team can pick up the work in the tool they already use.
- **Context files and memory.** Upload organization-specific context files. The agent applies them to its answers and remembers your preferences across sessions.

For how to use each feature, see Chatting with AWS FinOps Agent, Task management, and Context files and memory.

Related services

AWS FinOps Agent integrates with AWS services that you permit during the agent creation process. For the full list of IAM actions, see the AWS FinOps Agent IAM setup guide.

- **AWS Cost Explorer**. Cost and usage data retrieval, forecasting, Savings Plans and Reserved Instance analysis.
- **AWS Cost Anomaly Detection**. Anomaly monitoring and detection.
- **AWS Cost Optimization Hub**. Optimization recommendations and savings estimates.
- **AWS Compute Optimizer**. Detailed rightsizing and resource recommendations.
- **AWS CloudTrail**. API activity logs for identifying infrastructure changes during anomaly investigations.

To get started, see Creating an agent.

Getting started with AWS FinOps Agent

To start using AWS FinOps Agent, sign in to the AWS Management Console, switch to the US East (N. Virginia) (us-east-1) Region, and open the AWS FinOps Agent Console page. Create your first agent with the creation wizard. The wizard creates the IAM roles the agent needs and attaches the required policies for you, so you can get started without configuring IAM manually. You can optionally connect Jira and Slack during creation so the agent can create tickets and post messages.

After the agent is created, open its web application to upload your initial context (an account-to-owner mapping and any organization-specific instructions such as known exceptions, prioritization rules, and review cadence), run your first query, and set up your first event-triggered cost anomaly detection automation. The agent acts only on the data sources and integrations you connect during setup.

If your IAM administrator manages permissions centrally, or if you want to author the agent's roles and policies yourself, see the AWS FinOps Agent IAM setup guide for the full list of IAM policies, roles, and trust relationships.

Creating an agent

This topic walks through creating a AWS FinOps Agent agent in the AWS FinOps Agent console. The wizard has five steps: name your agent, choose what AWS resources the agent can access, give the web app access to your agent, third-party integrations (optional), and review and create.

A FinOps agent is an independent instance of AWS FinOps Agent that you configure in your AWS account. Each agent operates independently with its own IAM permissions, context files, memory, task queue, and third-party integrations. No data, resources, or permissions are shared across agents.

Prerequisites

Verify that you have the following in place:

- **Administrator setup permissions.** Your IAM identity needs permissions to create the agent and, when you choose **Auto-create a new role**, to create the agent and operator service roles. These permissions are defined in the `FinOpsAgentSetupPolicy`.

- **Existing service roles (optional).** The wizard can create the agent's roles for you. If you would rather provide your own, prepare a role for the agent (with `FinOpsAgentAgentPolicy`) and a role for the operator (with `FinOpsAgentOperatorPolicy`). Both roles must trust `finops-agent.amazonaws.com`. For the trust policy, see Trust policy.
- **Third-party integrations (optional).** For Jira, see Enable Jira with AWS FinOps Agent. For Slack, see Enable Slack with AWS FinOps Agent. You can skip this step during agent creation and add integrations later from the agent detail page.

The creation wizard

On the Agents page in the AWS FinOps Agent console, choose **Create** to launch the wizard.

Step 1: Name your agent

Enter a name and an optional description.

- **Agent name.** Letters, numbers, spaces, and hyphens only. Up to 128 characters. Underscores, dots, and other special characters are not allowed.
- **Description (optional).** Letters, numbers, spaces, and common punctuation (`- . , ! ? ; : ' " ( ) & /`). Up to 512 characters.

Step 2: Choose what AWS resources the agent can access

Choose the IAM role that AWS FinOps Agent assumes to read your AWS cost and operational data, and to manage event-based automations. You have two options.

Option 1: Auto-create a new role (recommended). The wizard creates the agent role with `FinOpsAgentAgentPolicy` attached and a trust policy scoped to your account and to this agent. No manual IAM configuration is required.

Option 2: Use an existing role. Select a role you have already created. The role must have `FinOpsAgentAgentPolicy` (or an equivalent policy) attached, and must trust `finops-agent.amazonaws.com`. For the required actions and trust policy, see `FinOpsAgentAgentPolicy` and Trust policy.

Step 3: Give the web app access to your agent

Choose the IAM role that AWS FinOps Agent assumes to run web application operations such as conversations, tasks, automations, and document management. The two options match Step 2.

Option 1: Auto-create a new role (recommended). The wizard creates the operator role with `FinOpsAgentOperatorPolicy` attached.

Option 2: Use an existing role. Select a role you have already created. The role must have `FinOpsAgentOperatorPolicy` (or an equivalent policy) attached, and must trust `finops-agent.amazonaws.com`.

Step 4: Third-party integrations (optional)

Optionally connect Jira and Slack to this agent. Both selections require that you have already installed the integration at the account level. If you have not, skip this step and add connections later from the agent detail page.

Connect with Jira

Authorize the agent to create Jira issues for cost anomalies, optimization recommendations, and other actionable findings you ask it to act on.

- **Jira integration.** Select an existing Jira integration registered in this account.
- **Space key.** Enter the Jira project space key the agent should use. Two to ten uppercase letters, starting with a letter (for example, `ENG`, `KAN`, or `FIN`).

For the integration setup process, see Enable Jira with AWS FinOps Agent.

Connect with Slack

Authorize the agent to post task and automation results to Slack channels you choose.

- **Slack integration.** Select an existing Slack integration registered in this account.
- **Channel ID.** Enter a Slack channel ID. Nine to twelve uppercase alphanumeric characters (for example, `C04ABCDEF12`). To find the channel ID in Slack, right-click the channel name, choose **View channel details**, and copy the ID at the bottom. Before you connect, add the AWS FinOps Agent Slack app to the channel.

For the integration setup process, see Enable Slack with AWS FinOps Agent.

Step 5: Review and create

Review the summary of your agent name, what AWS resources the agent can access, web app access, and integration selections. Choose **Edit** on any step to make changes.

Complete agent creation

When the configuration is correct, choose **Create agent**. AWS FinOps Agent creates the agent, provisions any auto-created roles, attaches the integration connections, and generates the web application.

Verifying your agent setup

After creation, the agent appears on the Agents page in the AWS FinOps Agent console. Open the agent detail page to confirm the following:

- The agent role and operator role are listed under **Permissions**.
- Any Jira or Slack connections you added in Step 4 appear under **Integrations**.
- Choose **Open agent** to open the web application in a new browser tab. The web application authenticates through your AWS console session.

Accessing the web application

After the agent is created, access the web application from the AWS FinOps Agent console. Navigate to the Agents page and locate your agent. You have two ways to open the web application:

- Choose **Open** in the **Open agent** column of the Agents table.
- Choose the agent name to open the agent detail page, then choose **Open agent**.

The web application opens in a new browser tab.

Sharing access with your team

To give other users access to the agent, share the AWS account ID and the agent name. Users navigate to the AWS FinOps Agent console in the same AWS account, locate the agent by name, and launch the web application from there. Each user authenticates through their own AWS console session. Users need the end-user web app policy attached to their IAM identity.

Managing your agent after creation

From the agent detail page in the console, you can do the following:

- View the agent configuration, including IAM roles and integration connections.
- Connect or disconnect Jira projects and Slack channels.
- Access the web application.
- View and manage historical chat conversations in the web application.
- Delete the agent.

Before you can delete an agent, you must delete both its Jira and Slack connections. Open the agent detail page, choose the **Integrations** tab, and delete each connection.

To delete an agent, select it on the Agents page and choose **Delete**. You can select multiple agents for bulk deletion. After deletion, the web application link and previous interaction data are removed.

Next steps

- Learn how to interact with the agent in the web application, including chatting, creating tasks, and uploading context files.
- Configure Jira or configure Slack if you skipped these during setup.

Using the web application

The web application is where you do day-to-day work with AWS FinOps Agent. From it, you can chat with the agent, manage tasks and automations, upload context files, and download generated artifacts.

Accessing the web application

Open the AWS FinOps Agent console in your AWS account, navigate to the Agents page, and locate your agent. Choose the web application link to open it in a new browser tab. You authenticate through your AWS console session. During preview, each session lasts 30 minutes. After the session expires, you are redirected to re-authenticate through the AWS console.

> **Note**
>
> **Trouble accessing the web application?** If you cannot open the web application or receive a permissions error, check the following:

1. Verify with your administrator that your IAM identity has the AWS FinOps Agent operator permissions attached. Without these permissions, the web application cannot authenticate your session. See End-user web app policy for the required policy.
2. Check whether your AWS console session has expired. If it has, sign in to the AWS console again and reopen the web application from the AWS FinOps Agent console.

Layout overview

The web application has three sections: a **side navigation** on the left, a **chat area** in the center, and a **workspace panel** on the right that opens when you choose a side navigation item.

- **Side navigation.** Switches between workspaces and tracks your past conversations. See Side navigation.
- **Chat area.** The primary way to interact with the agent. See Chat.
- **Workspace panel.** Opens to show the workspace you choose from the side navigation: Tasks, Automations, Artifacts, or Context files. See Workspaces.

Side navigation

The side navigation on the left has three sections: a **New chat** button at the top, a list of **workspace links**, and a **Recent** conversation list.

- **New chat.** Starts a new conversation while keeping the workspace panel you have open. Use this to begin a new topic without losing the page you were on.
- **Tasks.** Opens the Tasks workspace, where you view and manage on-demand tasks the agent runs in the background.
- **Automations.** Opens the Automations workspace, where you create, view, enable, disable, or delete recurring and event-triggered automations.
- **Artifacts.** Opens the Artifacts workspace, where you download files the agent generated, such as HTML, PDF, and PPT reports.
- **Context files.** Opens the Context files workspace, where you upload, soft-delete, and restore files the agent uses to understand your organization.
- **Recent.** Lists your past conversations sorted by last activity. Choose a conversation to reopen it. Choose **Load more** at the bottom of the list to fetch additional pages.

The agent's display name appears in the bottom of the side navigation, so you can confirm which agent you are using when your team has more than one agent.

Chat

The chat area is the primary way to interact with the agent. The initial view shows a prompt input and a set of suggested prompts under **Prompts to try out**. Choose a suggested prompt or type your own message and choose send. Each chat message can be up to 1,000 characters.

For an overview of what you can ask, see Chatting with AWS FinOps Agent.

Continuing a conversation. Each conversation maintains its own context. You can ask follow-up questions that reference earlier parts of the conversation. If you navigate to a workspace and return, your conversation is preserved.

Starting a new conversation. Choose **New chat** at the top of the side navigation. Your previous conversation remains accessible from the **Recent** list.

Reopening a past conversation. Choose any conversation in the **Recent** list to load it and continue where you left off.

Workspaces

Each link in the side navigation opens a workspace in the right panel. The four workspaces are described in this section.

Tasks. Lists all tasks sorted by last updated time. Filter by **All tasks**, **Awaiting approval**, **In progress**, or **Completed**. Recently created tasks appear at the top. Use the refresh button to show updated status. Each row shows the task ID, name, status, creation time, and last updated time. Choose a task to open its detail page, which includes the overview, instructions, execution activities, generated files, and status. From the task detail page, you can cancel a running task. For the task lifecycle, see Task management.

Automations. Lists recurring and event-triggered automations. An automation defines what the agent does and when it runs. From this workspace, you can view all automations with their trigger type, schedule, status, and last triggered time; create a new automation with instructions and trigger configuration; and delete an automation. To enable or disable an automation without deleting it, open its detail page. Existing tasks already created by an automation are not affected when the automation is deleted.

Artifacts. Lists files the agent has generated, such as HTML, PDF, and PPT reports and analysis results. Each row shows the file name, type, size, and creation time. Choose an artifact to download the file or to open the source task that produced it.

Context files. Lists files you have uploaded to help the agent understand your organization. From this workspace, you can view all context files with their name, type, size, status, and upload time; upload a new context file (for supported types and limits, see Context files and memory); and soft-delete or restore files.

Approvals

When the agent needs to create a Jira issue or add a comment to one, it pauses and requests your approval before proceeding. Pending approvals appear in the Tasks workspace under the **Awaiting approval** filter. Open the task detail page to see the proposed action, then approve or reject it.

For chat-initiated tasks, the agent asks for approval in the conversation. For scheduled and event-based automations that include Jira issue creation or comment addition, no approval is required. When you set up the automation, you pre-authorize the agent to take those actions each time the automation runs.

For details on guardrail behavior, see Agent guardrail controls.

Providing feedback

The simplest way to give feedback is through the thumbs up and thumbs down buttons in the web application. Use the channels in this section for documentation issues, larger asks, and account-team escalations.

In-product feedback (thumbs up and thumbs down)

Every agent response and every task result has thumbs up and thumbs down buttons. Use them to tell the AWS FinOps Agent team whether a specific response or task result was useful.

- **In a chat conversation.** The buttons appear after each agent response. Choose them to rate that turn.
- **On a task result.** The buttons appear at the bottom of the task detail page. Choose them to rate the overall result of the task.

When you choose **thumbs up**, the rating is submitted immediately. No comment is required.

When you choose **thumbs down**, a dialog opens so you can explain what was wrong. Select one or more categories and add an optional free-text comment.

Category	When to choose
Harmful	The response contained content that was harmful, unsafe, or violated policy.
Not accurate	The numbers, claims, or analysis were factually wrong.
Not useful or incomplete	The response missed the question, was too vague, or stopped short of what was asked.
Something else	None of the categories above fits. Use the comment field to describe the issue.

Feedback you submit is associated with the conversation turn or task you rated.

Documentation feedback

To submit feedback about this user guide (corrections, missing information, suggestions), choose **Feedback** at the bottom of any page. Documentation feedback is routed to the AWS FinOps Agent documentation team.

AWS FinOps Agent IAM setup guide

When you create an agent, the creation wizard can create the required IAM roles and attach the policies for you. Most customers do not need to configure IAM manually. For the standard flow, see Creating an agent.

Use this topic if your IAM administrator manages permissions centrally, or if you want to author the roles and policies yourself. AWS FinOps Agent uses four IAM policies and two IAM roles. This topic walks through each policy, the roles that the policies attach to, and how to enable the AWS services that the agent depends on.

Step 1: Create IAM policies

Create the following four IAM policies. The policy names shown are samples and can be customized.

Policy 1: Admin setup policy

Sample name: `FinOpsAgentSetupPolicy`

This policy grants the administrator permissions to create and manage AWS FinOps Agent instances, configure third-party integrations, and generate login sessions for web application users. Attach this policy directly to the administrator's IAM user or role.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FinOpsAgentAdminSetup",
      "Effect": "Allow",
      "Action": [
        "finops-agent:CreateAgentSpace",
        "finops-agent:GetAgentSpace",
        "finops-agent:ListAgentSpaces",
        "finops-agent:UpdateAgentSpace",
        "finops-agent:DeleteAgentSpace",
        "finops-agent:CreateConnection",
        "finops-agent:GetConnection",
        "finops-agent:ListConnections",
        "finops-agent:UpdateConnection",
        "finops-agent:DeleteConnection",
        "finops-agent:CreateIntegration",
        "finops-agent:GetIntegration",
        "finops-agent:ListIntegrations",
        "finops-agent:DeleteIntegration",
        "finops-agent:CreateOneTimeLoginSession"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamReadForRolePicker",
      "Effect": "Allow",
      "Action": ["iam:GetRole", "iam:ListRoles"],
```

```
      "Resource": "*"
    },
    {
      "Sid": "CreateFinOpsServiceRolesOnly",
      "Effect": "Allow",
      "Action": "iam:CreateRole",
      "Resource": "arn:aws:iam::*:role/service-role/*"
    },
    {
      "Sid": "AttachOnlyFinOpsManagedPolicies",
      "Effect": "Allow",
      "Action": "iam:AttachRolePolicy",
      "Resource": "arn:aws:iam::*:role/service-role/*",
      "Condition": {
        "ArnEquals": {
          "iam:PolicyARN": [
            "arn:aws:iam::aws:policy/FinOpsAgentAgentPolicy",
            "arn:aws:iam::aws:policy/FinOpsAgentOperatorPolicy"
          ]
        }
      }
    },
    {
      "Sid": "PassFinOpsRolesToService",
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "finops-agent.amazonaws.com"
        },
        "ArnLike": {
          "iam:AssociatedResourceArn": "arn:aws:finops-agent:*:*:agentspace/*"
        }
      }
    }
  ]
}
```

Policy 2: Agent permissions policy

Sample name: `FinOpsAgentAgentPolicy`

This policy defines what AWS services and data the AWS FinOps Agent can read in your account. The agent uses these permissions to query billing and cost data, retrieve optimization recommendations, look up infrastructure details, and correlate cost changes with operational metrics. This policy will be attached to the agent IAM role in Step 2.

You have two options for creating this policy:

- **Option 1: Auto-create during agent creation (recommended).** The agent creation wizard creates the policy as an AWS managed policy and attaches it to the agent role automatically. You can skip this section and let the wizard handle it.
- **Option 2: Author the policy manually.** Create the policy with the JSON below if your IAM administrator manages all permissions centrally. You will need to attach this policy to the agent IAM role in Step 2 manually.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FinOpsAgentDataAccess",
      "Effect": "Allow",
      "Action": [
        "ce:GetCostAndUsage",
        "ce:GetCostAndUsageWithResources",
        "ce:GetCostForecast",
        "ce:GetUsageForecast",
        "ce:GetDimensionValues",
        "ce:GetTags",
        "ce:GetCostCategories",
        "ce:GetCostAndUsageComparisons",
        "ce:GetCostComparisonDrivers",
        "ce:GetSavingsPlansCoverage",
        "ce:GetSavingsPlansUtilization",
        "ce:GetSavingsPlansUtilizationDetails",
        "ce:GetSavingsPlansPurchaseRecommendation",
        "ce:GetReservationCoverage",
        "ce:GetReservationUtilization",
        "ce:GetReservationPurchaseRecommendation",
        "ce:GetAnomalies",
        "ce:GetAnomalyMonitors",
        "ce:ListCostAllocationTags",
        "ce:ListCostAllocationTagBackfillHistory",
```

```
        "ce:DescribeCostCategoryDefinition",
        "ce:ListCostCategoryDefinitions",
        "budgets:ViewBudget",
        "cost-optimization-hub:GetRecommendation",
        "cost-optimization-hub:ListRecommendations",
        "cost-optimization-hub:ListRecommendationSummaries",
        "compute-optimizer:DescribeRecommendationExportJobs",
        "compute-optimizer:GetEnrollmentStatus",
        "compute-optimizer:GetEnrollmentStatusesForOrganization",
        "compute-optimizer:GetRecommendationSummaries",
        "compute-optimizer:GetEC2InstanceRecommendations",
        "compute-optimizer:GetEC2RecommendationProjectedMetrics",
        "compute-optimizer:GetAutoScalingGroupRecommendations",
        "compute-optimizer:GetEBSVolumeRecommendations",
        "compute-optimizer:GetLambdaFunctionRecommendations",
        "compute-optimizer:GetRecommendationPreferences",
        "compute-optimizer:GetEffectiveRecommendationPreferences",
        "compute-optimizer:GetECSServiceRecommendations",
        "compute-optimizer:GetECSServiceRecommendationProjectedMetrics",
        "compute-optimizer:GetLicenseRecommendations",
        "compute-optimizer:GetRDSDatabaseRecommendations",
        "compute-optimizer:GetRDSDatabaseRecommendationProjectedMetrics",
        "compute-optimizer:GetIdleRecommendations",
        "ec2:DescribeInstances",
        "ec2:DescribeVolumes",
        "ecs:ListServices",
        "ecs:ListClusters",
        "autoscaling:DescribeAutoScalingGroups",
        "autoscaling:DescribeAutoScalingInstances",
        "lambda:ListFunctions",
        "lambda:ListProvisionedConcurrencyConfigs",
        "organizations:ListAccounts",
        "organizations:DescribeOrganization",
        "organizations:DescribeAccount",
        "rds:DescribeDBInstances",
        "rds:DescribeDBClusters",
        "pricing:DescribeServices",
        "pricing:GetAttributeValues",
        "pricing:GetProducts",
        "freetier:GetFreeTierUsage",
        "bcm-pricing-calculator:GetPreferences",
        "bcm-pricing-calculator:GetWorkloadEstimate",
        "bcm-pricing-calculator:ListWorkloadEstimateUsage",
        "bcm-pricing-calculator:ListWorkloadEstimates",
```

```
        "cloudtrail:LookupEvents",
        "cloudtrail:DescribeTrails",
        "cloudtrail:GetTrailStatus",
        "cloudtrail:GetEventSelectors",
        "cloudwatch:GetMetricData",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:ListMetrics",
        "logs:StartQuery",
        "logs:GetQueryResults"
      ],
      "Resource": "*"
    },
    {
      "Sid": "EventBridgeManagedRuleManagementWritePermissions",
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets",
        "events:EnableRule",
        "events:DisableRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/*",
      "Condition": {
        "StringEquals": {
          "events:ManagedBy": "finops-agent.amazonaws.com",
          "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
      }
    },
    {
      "Sid": "EventBridgeManagedRuleManagementReadPermissions",
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceAccount": "${aws:PrincipalAccount}"
        }
      }
```

```
    }
  ]
}
```

AWS FinOps Agent uses AWS EventBridge actions to manage event-triggered automations for cost anomaly investigation. Without these permissions, the agent cannot create event-triggered automations for cost anomaly investigation. The `events:ManagedBy` condition restricts the agent to managing only the EventBridge rules it created on your behalf, so rules you create directly are not affected. The `aws:ResourceAccount` condition limits the agent to managing rules in your own account.

The CloudTrail actions support cost anomaly investigation. The agent uses `cloudtrail:LookupEvents` to find the API activity behind a cost change.

You can remove actions you do not need. For example, if you remove `cloudtrail:LookupEvents`, the agent continues to work for cost inquiry, reporting, and recommendations, and it still detects and analyzes cost anomalies, but it can no longer correlate a cost spike with the CloudTrail records that explain what changed.

Policy 3: Operator permissions policy

Sample name: `FinOpsAgentOperatorPolicy`

This policy defines what actions the web application can perform with the AWS FinOps Agent service. It covers managing conversations, tasks, automations, context files, and reports. This policy will be attached to the operator IAM role in Step 2.

You have two options for creating this policy:

- **Option 1: Auto-create during agent creation (recommended).** The agent creation wizard creates the policy as an AWS managed policy and attaches it to the operator role automatically. You can skip this section and let the wizard handle it.
- **Option 2: Author the policy manually.** Create the policy with the JSON below if your IAM administrator manages all permissions centrally. You will need to attach this policy to the operator IAM role in Step 2 manually.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "FinOpsAgentOperatorAccess",
      "Effect": "Allow",
      "Action": [
        "finops-agent:CreateConversation",
        "finops-agent:ListConversations",
        "finops-agent:CreateTurn",
        "finops-agent:GetTurn",
        "finops-agent:ListTurns",
        "finops-agent:CancelTurn",
        "finops-agent:AcceptAgentRequest",
        "finops-agent:RejectAgentRequest",
        "finops-agent:GetAgentRequest",
        "finops-agent:CreateTask",
        "finops-agent:GetTask",
        "finops-agent:ListTasks",
        "finops-agent:CancelTask",
        "finops-agent:CreateAutomation",
        "finops-agent:GetAutomation",
        "finops-agent:ListAutomations",
        "finops-agent:UpdateAutomation",
        "finops-agent:DeleteAutomation",
        "finops-agent:CreateDocument",
        "finops-agent:GetDocumentContent",
        "finops-agent:GetDocumentMetadata",
        "finops-agent:ListDocuments",
        "finops-agent:UpdateDocument",
        "finops-agent:DeleteDocument",
        "finops-agent:RestoreDocument",
        "finops-agent:DeleteArtifact",
        "finops-agent:GetArtifactContent",
        "finops-agent:GetArtifactMetadata",
        "finops-agent:ListArtifacts",
        "finops-agent:ListRecords",
        "finops-agent:SendFeedback"
      ],
      "Resource": "*"
    }
  ]
}
```

Policy 4: Web app user policy

Sample name: `FinOpsAgentWebAppPolicy`

This policy grants your team members permissions to find the agent in the AWS console and create a login session for the web application. Attach this policy directly to each team member's IAM user or role.

This policy does not allow users to create or delete agents. To allow a user to create or delete agents, attach Policy 1 (`FinOpsAgentSetupPolicy`) to that user or role.

The Cost Explorer read actions in this policy are optional and not required to access the web application. They enable users to cross-validate the agent's findings by viewing cost data directly in Cost Explorer within the same console session. If your team does not need this, you can remove the `ce:*` actions from the policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FinOpsAgentUsers",
      "Effect": "Allow",
      "Action": [
        "finops-agent:GetAgentSpace",
        "finops-agent:ListAgentSpaces",
        "finops-agent:ListConnections",
        "finops-agent:GetConnection",
        "finops-agent:ListIntegrations",
        "finops-agent:CreateOneTimeLoginSession",
        "iam:GetRole",
        "ce:GetCostAndUsage",
        "ce:GetCostAndUsageWithResources",
        "ce:GetCostForecast",
        "ce:GetUsageForecast",
        "ce:GetDimensionValues",
        "ce:GetTags",
        "ce:GetCostCategories",
        "ce:GetCostAndUsageComparisons",
        "ce:GetCostComparisonDrivers",
        "ce:GetSavingsPlansCoverage",
        "ce:GetSavingsPlansUtilization",
        "ce:GetSavingsPlansUtilizationDetails",
        "ce:GetSavingsPlansPurchaseRecommendation",
        "ce:GetReservationCoverage",
        "ce:GetReservationUtilization",
        "ce:GetReservationPurchaseRecommendation"
      ],
```

```
      "Resource": "*"
    }
  ]
}
```

Step 2: Create two IAM roles

AWS FinOps Agent requires two IAM roles. Each role serves a different purpose and is assumed by the AWS FinOps Agent service to perform different operations.

The **agent role** (sample name: `FinOpsAgentRole`) is the role the AWS FinOps Agent service assumes to query your AWS billing data, optimization recommendations, and infrastructure metrics. When you ask the agent a question about your costs, the service uses this role to call AWS APIs like AWS Cost Explorer and AWS Compute Optimizer. Attach Policy 2 (`FinOpsAgentAgentPolicy`) to this role.

The **operator role** (sample name: `FinOpsAgentOperatorRole`) is the role the AWS FinOps Agent service assumes to perform web application operations. When you send a chat message, create a task, or upload a context file, the service uses this role's credentials to execute those actions. Attach Policy 3 (`FinOpsAgentOperatorPolicy`) to this role.

Create two IAM roles using the trust policy below, then attach the corresponding permissions policy from Step 1.

IAM role (sample name)	Attach this policy from Step 1
`FinOpsAgentRole`	Policy 2 (`FinOpsAgentAgentPolicy`)
`FinOpsAgentOperatorRole`	Policy 3 (`FinOpsAgentOperatorPolicy`)

Trust policy

Both roles use the same trust policy. This trust policy allows the AWS FinOps Agent service account to assume the role using `sts:AssumeRole` and to stamp the calling user's identity onto the session using `sts:SetSourceIdentity`.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "finops-agent.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetSourceIdentity"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "{{accountId}}"
        },
        "ArnLike": {
          "aws:SourceArn": "arn:aws:finops-agent:*:{{accountId}}:agentspace/*"
        }
      }
    }
  ]
}
```

Replace `{{accountId}}` with your AWS account ID. The `agentspace/*` wildcard allows any agent in the account to assume the role. To restrict the role to a specific agent, replace the wildcard with the agent ID after you create the agent.

Why `sts:SetSourceIdentity`? When AWS FinOps Agent assumes one of these roles on behalf of a web application user, it sets the user's IAM unique identifier as the session's `SourceIdentity`. Every AWS CloudTrail event the agent generates while using these credentials carries that `sourceIdentity` field, so you can attribute agent-driven activity back to the individual user. Without this permission, the role assumption still succeeds, but downstream CloudTrail events do not include the `sourceIdentity` field. For details, see How caller identity appears in CloudTrail.

Activate dependent AWS services

After you create the IAM policies and roles, activate the underlying AWS services that the agent uses. IAM permissions alone are not enough for the cost optimization and cost anomaly investigation features.

AWS Compute Optimizer (for cost optimization recommendations)

Open the AWS Compute Optimizer console and opt in. Without opt-in, the agent cannot retrieve rightsizing or idle resource recommendations from AWS Compute Optimizer. For details, see Getting started with AWS Compute Optimizer.

AWS Cost Anomaly Detection (for cost anomaly investigation)

Open the AWS Cost Anomaly Detection page in the console and create at least one anomaly monitor. The agent investigates anomalies that AWS Cost Anomaly Detection produces from your monitors. For details, see Getting started with AWS Cost Anomaly Detection.

AWS Cost Optimization Hub (for cost optimization recommendations)

AWS Cost Optimization Hub is enabled by default in every AWS account. No additional setup is required.

AWS CloudTrail (for cost anomaly investigation)

The agent uses CloudTrail Event History (through `LookupEvents`) to identify the change behind a cost spike. CloudTrail Event History is enabled by default in every AWS account at no charge. You do not need to create a trail or configure CloudTrail for the agent to investigate anomalies.

Working with AWS FinOps Agent

After your agent is created, you interact with it through chat in the web application. This chapter describes what you can do with the agent and how to operationalize it. Start with the use cases to see what the agent can do from chat, then set up an automated workflow that runs without manual triage. The remaining topics cover the task queue and automations, teaching the agent about your business through context files and memory, and enabling Jira and Slack.

The use cases show what the agent can do from a single conversation: ask about costs, investigate anomalies, surface optimization recommendations, and generate reports. Any of these can run on a schedule or in response to an event. For a complete hands-off example, event-triggered cost anomaly investigation connects a AWS Cost Anomaly Detection trigger, an investigation, and a Slack destination into one workflow.

AWS FinOps Agent use cases

The chat area in the web application is the primary way to interact with AWS FinOps Agent. From a single conversation, you can ask cost questions in plain language and get answers using your cost and usage data, investigate cost anomalies to root cause, surface optimization recommendations, and generate reports. The agent reads your context files, applies what it has remembered from previous sessions, and uses connected integrations such as Jira and Slack to create a Jira ticket or post results to a Slack channel.

The following topics walk through each use case, with sample prompts. You can also turn any of these into recurring or event-driven work. For the task and automation model, see Task management. For an end-to-end automation walkthrough, see Event-triggered cost anomaly investigation.

> **Note**
>
> **Pricing for underlying AWS API calls.** AWS FinOps Agent is offered at no charge during preview, but the agent calls AWS APIs on your behalf and you pay the standard per-request rate for those APIs. For details, see pricing. Other services the agent reads from, including Cost Anomaly Detection, Cost Optimization Hub, Compute Optimizer, and CloudTrail Event History, are available at no additional charge.

Asking about costs

Ask questions about your AWS spend in natural language. The agent retrieves data from services, processes the results, and responds with answers based on your account data. The agent can break down spend by service, account, Region, or tag, compare time periods, and summarize Savings Plans and Reserved Instance utilization.

The agent accesses the following services through its configured IAM role:

Data source	What it provides
AWS Cost Explorer	Cost and usage data with flexible grouping and filtering, cost forecasts, usage forecasts , dimension values, tags, cost categorie s, Savings Plans coverage and utilization and purchase recommendations, Reserved Instance coverage and utilization and purchase recommendations.
AWS Cost Anomaly Detection	Detected anomalies and anomaly monitors.
AWS Cost Optimization Hub	Cost-saving recommendations and recommendation summaries across services.
AWS Compute Optimizer	Rightsizing recommendations for EC2, Auto Scaling groups, EBS, Lambda, RDS, ECS, and idle resource detection.

The depth of the agent's responses depends on which IAM permissions are enabled. If a data source is not granted, the agent disables the tools that depend on it and notes the limitation in its response. For the full list of IAM actions, see the AWS FinOps Agent IAM setup guide.

You can ask follow-up questions within the same conversation to refine results or explore related topics. The agent retains context across sessions through memory and applies preferences you have explicitly recorded, such as preferred cost views or common breakdowns, in future interactions.

Sample prompts:

- “Summarize our AWS costs for last month compared to the month before.”
- “What were our top 5 cost drivers last month? Break it down by service.”
- “Which services had the biggest increase?”
- “Show me EC2 costs by Region for the last 3 months.”
- “What's our month-over-month spend trend?”
- “Show me spend by linked account.”
- “Show me costs grouped by cost-center tag.”
- “What's the cost forecast for next month?”

Investigating cost anomalies

Ask the agent to investigate why AWS Cost Anomaly Detection anomalies occurred and perform root-cause analysis. The agent retrieves the anomaly details, analyzes the root causes ranked by cost impact, and presents its findings directly in the conversation.

The agent determines whether cost increases were driven by higher usage, higher rates, or both. When CloudTrail permissions are enabled, the agent searches CloudTrail event history for API activity correlated with the cost change.

The depth of root-cause analysis depends on which permissions are enabled in the agent's IAM role:

Permissions enabled	Investigation depth
Cost Explorer only	Anomaly details with affected services, accounts, cost impact, cost trends, and rate-vs-usage analysis.
Cost Explorer and CloudTrail	Full investigation including the above, plus a search of CloudTrail event history for API activity correlated with the cost change.

You can ask the agent to save the investigation as a downloadable file, create a Jira ticket with the summary, or post the result to a Slack channel.

To investigate anomalies automatically whenever AWS Cost Anomaly Detection detects one, without starting a conversation each time, see Event-triggered cost anomaly investigation.

Sample prompts:

- “Have there been any cost anomalies in the past 7 days? Investigate them.”
- “Why did our costs jump this week?”
- “Tell me more about the MemoryDB anomaly. What happened?”
- “When a cost anomaly is detected on my production monitor, investigate the root cause and post the summary to *<slack-channel>*.”
- “Set up monitoring so I'm alerted if this happens again at the same threshold, and auto-investigate.”

Event-triggered cost anomaly investigation

This topic walks through setting up an end-to-end event-triggered workflow: the agent listens for AWS Cost Anomaly Detection events, investigates each anomaly for root cause, and posts the summary to a Slack channel. After you set this up once, anomalies are investigated and delivered to your team without manual triage.

The workflow combines three pieces:

- **A trigger** — a AWS Cost Anomaly Detection anomaly event.
- **An action** — an investigation that the agent runs when the event arrives.
- **A destination** — the Slack channel the agent posts the summary to.

Prerequisites

- At least one anomaly monitor in AWS Cost Anomaly Detection. The agent investigates anomalies that your monitors produce. To create a monitor, see Getting started with AWS Cost Anomaly Detection.
- The agent role has the AWS EventBridge permissions in `FinOpsAgentAgentPolicy`. The agent uses these to create the managed rule that receives anomaly events. If you created the agent with the wizard's auto-created role, these permissions are already in place.
- A Slack channel where the agent posts results, connected as described in Step 1.

Step 1: Connect a Slack channel

Connect the Slack channel that the agent posts investigation summaries to. If you already connected the channel during agent creation, skip to Step 2.

Connecting Slack to an agent is a two-part process: register a Slack integration at the account level, then create a connection that binds a channel to your agent.

1. Register a Slack integration for your account, if you have not already. This is a one-time setup per Slack workspace. For the full process, including adding the AWS FinOps Agent Slack app to the channel, see Enable Slack with AWS FinOps Agent.
2. Open your agent from the AWS FinOps Agent console, where you will see an **Add connection** button.
3. Choose **Add connection**, then choose **Slack**. The Slack option is available only after a Slack integration is registered in your account.
4. Select the Slack integration, enter the channel ID of the channel the agent posts to, then choose **Create**.

Step 2: Describe the trigger and response in chat

In the chat area, send a prompt that describes the event you want the agent to listen for and what it should do when the event arrives. State the monitor or scope, the investigation, and the Slack channel in a single prompt. The agent recognizes a prompt like this as a request to set up ongoing automation and creates the event-based automation directly, rather than running a one-time investigation.

For example, either of these prompts creates an event-based automation:

- *"When a cost anomaly is detected on my production monitor, investigate the root cause and post the summary to `<slack-channel>`."*
- *"When a cost anomaly over $1,000 is detected, investigate the root cause and post the findings to `<slack-channel>`."*

Include a filter in the prompt, such as a dollar threshold, to narrow what the agent acts on so your team's attention stays on the highest-impact changes. The second example above investigates only anomalies over $1,000.

From your prompt, the agent creates an event-based automation with a `COST_ANOMALY` trigger. The automation stores the prompt you described and the destination. Each time AWS Cost Anomaly Detection emits a matching anomaly event, the agent creates and runs a task from the automation, investigates the anomaly, and posts the summary to the Slack channel you specified.

Behind the scenes, the agent provisions an AWS EventBridge managed rule in your account to receive the anomaly events. The rule is scoped so that the agent manages only the rules it created. For the underlying permissions, see `FinOpsAgentAgentPolicy`.

> **Note**
>
> Posting to Slack does not require approval, so the automation runs end to end without prompting. If your automation also creates a Jira issue, that action is pre-authorized when you set up the automation, so it also runs without a per-event approval prompt. For the full approval model, see Agent guardrail controls.

Step 3: Verify and manage the automation

Open the **Automations** workspace in the web application to confirm the automation was created. The workspace lists each automation with its trigger type, status, and last triggered time.

From the automation detail page, you can do the following:

- Enable or disable the automation without deleting it.
- Delete the automation. Deleting it stops new tasks from being created on the trigger; tasks already created remain on the agent.
- Review the tasks the automation has run, including each investigation and its delivery result.

To investigate an anomaly on demand instead of automatically, see Investigating cost anomalies.

Surfacing cost optimization recommendations

Ask the agent to retrieve, filter, and present cost-saving opportunities from AWS Cost Optimization Hub and AWS Compute Optimizer. Each recommendation includes the affected resource, current and recommended configuration, estimated monthly savings, implementation effort, and implementation guidance.

The agent retrieves recommendations from two complementary sources. **AWS Cost Optimization Hub** aggregates optimization opportunities across your account and groups them by action type (rightsizing, idle resource deletion, Savings Plans, Reserved Instances, architecture migration). **AWS Compute Optimizer** provides resource-level recommendations for EC2, Auto Scaling groups, EBS, Lambda, RDS, and ECS, including utilization metrics and per-resource savings.

The agent presents several categories of opportunities:

- **Rightsizing.** Resources overprovisioned relative to actual usage, with current configuration, recommended smaller configuration, and supporting utilization data.
- **Idle resources.** Unused resources that can be stopped or deleted, including unattached EBS volumes, idle EC2 instances, and unused DynamoDB tables and global secondary indexes.
- **Savings Plans and Reserved Instances.** Opportunities to reduce costs by committing to consistent usage, with current coverage and estimated savings from additional commitments.
- **Architecture migration.** Opportunities to move to more cost-effective platforms, such as migrating EC2 instances to Graviton processors or converting GP2 EBS volumes to GP3.

You can filter recommendations by account, Region, service, action type, effort level, or minimum savings threshold. By default, the agent orders recommendations by estimated monthly savings, highest first. You can use context files to define rules such as ignoring rightsizing in production accounts, discarding recommendations below a savings threshold, or grouping by engineering team.

From the agent, you can summarize selected recommendations into a Jira ticket. The engineering team picks up the work in Jira. Each ticket includes the recommendation summary, estimated savings, affected resources, effort level, and implementation guidance. For recurring optimization reviews, create a scheduled automation that reviews new recommendations on a cadence you set and summarizes them into Jira tickets. For details on scheduled automations, see Task management.

Sample prompts:

- “Show me all optimization opportunities sorted by savings.”
- “Narrow it to low-effort ones saving over $500 per month.”
- “What are our top quick-win optimization opportunities?”
- “What instance type should I downsize i-abc123 to?”

- “Estimate my annual savings if I implement all recommendations.”
- “Every Monday, review new optimization opportunities and create Jira tickets in *<jira-space-key>*.”

Generating cost reports

Ask the agent to generate financial reports in natural language. The agent creates HTML, PDF, and PPT reports with cost summaries, breakdowns by service, account, Region, or tag, trend analyses, month-over-month comparisons, and forecast projections. If you don't specify a format, the agent defaults to HTML.

Describe the report you want. The agent retrieves data from Cost Explorer and Cost Anomaly Detection, processes the data, builds charts, and assembles the report in the format you request. Reports include visual elements such as charts, stat callouts, comparison columns, and icon-annotated sections rather than plain text and bullet points. The agent runs a quality assurance pass that inspects the output for visual issues before delivering the final file. You can also upload sample reports as context files for the agent to replicate format and style.

Report formats. Choose the format that matches your audience and how the report will be consumed.

Format	When to use it
HTML (`.html`)	Default. Best for browser viewing and for quickly sharing a link to the artifact. Charts render in the browser.
PDF (`.pdf`)	Best for archival, email attachments, and reports that need to print well. Layout is fixed across viewers.
PPT (`.pptx`)	Best for executive reviews and walkthroughs. The agent produces an editable slide deck so you can rearrange, retitle, or annotate slides before presenting.

Reports are delivered as artifacts in the agent's response. Download the report from the **Artifacts** workspace in the web application, send it to a Slack channel through a connected Slack integration, or reference it in a follow-up turn so the agent can revise it without regenerating from scratch.

For recurring reporting needs, create a scheduled task that generates and delivers reports on a defined cadence. For details on scheduled tasks, see Task management.

Sample prompts:

- “Generate an HTML cost report summarizing our cost trends, top cost drivers, and optimization opportunities. I want to share this with my CFO.”
- “Generate a PowerPoint covering last month's costs by service, including the anomalies.”
- “Create a PDF report of last month's spend by linked account.”
- “Take the weekly report and add a section for anomalies.”
- “Every Monday at 9 AM, generate a cost summary for the VP of Engineering and post it to *<slack-channel>*.”

Task management

AWS FinOps Agent supports three task types: on-demand, scheduled, and event-based. Each runs through a shared task queue.

On-demand tasks

On-demand tasks run in the background without blocking the conversation. When you ask the agent to create a task, it runs the work in the background and continues the conversation. This is useful for requests that require extended processing, such as generating a report or investigating an anomaly. The agent creates a task only when you ask it to; it answers other requests directly in the conversation.

You can create on-demand tasks in two ways:

- **From the chat area.** Ask the agent to create a task in conversation. For example: “Create a task to generate a cost report for last month.” The agent creates the task and runs it in the background.
- **From the Tasks workspace.** In the side navigation, choose **Tasks**, then choose **Create tasks** to open the task creation form. The form includes instructions for how you want the task to be structured.

Scheduled tasks

Scheduled tasks run recurring workflows at specified intervals. Define the schedule through natural language (for example, “every Monday at 9 AM” or “on the first business day of each month”) and the agent creates a recurring automation in its queue. Each time the schedule triggers, the agent creates and runs a new task from the automation.

To view all automations, open the **Automations** workspace from the side navigation. To stop a recurring workflow, disable or delete the automation.

Event-based tasks

Event-based tasks trigger automated responses when conditions are met. During preview, the agent supports cost anomaly detection events from AWS Cost Anomaly Detection. When an anomaly is detected on a monitor that matches the event-based task, the agent runs the prompt you specified.

You configure event-based tasks by describing the trigger condition and desired response in conversation. For example: *“When a cost anomaly is detected on my production monitor, investigate the root cause and post the summary to my finance Slack channel.”* The agent registers an event-based automation that runs the prompt each time a matching event arrives.

Behind the scenes, the agent provisions an AWS EventBridge managed rule to receive the events. For details on the underlying IAM permissions, see `FinOpsAgentAgentPolicy`. For the end-to-end walkthrough, see Event-triggered cost anomaly investigation.

Approvals for read-write actions

During preview, the agent requires approval before creating a Jira issue or adding a comment to one, and only when the action is started from a chat conversation or from an on-demand task. The agent shows the action details (for an issue: project, summary, description; for a comment: target issue and body) in the conversation, and you confirm or revise before the action runs.

Automations that include Jira ticket creation (scheduled or event-based) do not prompt for approval. When you set up the automation, you pre-authorize the agent to create tickets each time the automation runs, so the workflow completes end to end without manual intervention.

Posting a message to a Slack channel does not require approval, including from a chat conversation or an on-demand task. The agent posts to the channel you specify and records the result in the task history. For the full read-write action model, see Agent guardrail controls.

All other agent actions (querying cost data, retrieving recommendations, posting to a Slack channel, generating reports) run without approval. You can review every action the agent took in the task history.

For details on guardrail policies, see Agent guardrail controls.

Reflect your organization with context files and memory

The agent uses two inputs to apply your organization's structure: context files that you upload, and memory that the agent records from your preference. The agent uses both automatically the next time it answers a question, generates a report, or runs an automation.

Apply your accounts, owners, and policies with context files

Context files let the agent answer using your organization's terms. Upload an account-to-owner mapping and the agent attributes spend to the right teams. Upload a cost category definition and the agent uses your categories in reports. Upload custom instructions and the agent follows them on every run. Without context files, the agent works from generic AWS data; with them, the agent's answers, reports, and tickets reflect your accounts, owners, and review cycles.

Start with an account-to-team mapping. This is the most useful context file to upload first, because it lets the agent attribute cost to the team that owns each account.

```
account_id,team_name,team_lead,email
123456789012,Data Platform,Jane Smith,jsmith@example.com
234567890123,ML Training,Alex Chen,achen@example.com
345678901234,Production,Sam Lee,slee@example.com
```

Commonly useful context files:

Context file	What it lets the agent do
Account-to-team mapping	Attribute spend to the team that owns each AWS account, and route Jira tickets to the responsible owner.
Organization structure	Allocate cost across business units, and explain reporting relationships in chargeback summaries.

Context file	What it lets the agent do
Custom instructions	Apply your rules every time. For example: "Ignore EC2 rightsizing recommendations for accounts tagged `env=production` " or "When creating Jira tickets, use the FINOPS project."
Company background	Calibrate the agent's communication style to your business and FinOps maturity level.
Report templates	Replicate the format and structure of reports your stakeholders already expect.

The web application accepts the file types listed in the next section. Context files are read-only from the agent's perspective; the agent cannot modify or delete them through conversation.

> **Important**
>
> Do not upload files that contain sensitive or personal information. Context files are visible to anyone who has access to the agent.

Supported file types and limits

The web application accepts the following file types for context file upload:

- Plain text (`.txt`)
- CSV (`.csv`)
- JSON (`.json`)
- Markdown (`.md`)
- HTML (`.html`)
- YAML (`.yaml`, `.yml`)

Files outside these types are rejected before upload.

The following limits apply per agent. For all per-agent quotas, see Quotas.

Limit	Value
Maximum file size per upload	10 MB
Total context file storage per agent	100 MB

Manage context files

Upload, delete, and restore context files from the **Context files** workspace in the web application. Choosing **Delete** on a file soft-deletes it, removing it from the agent's active context. Choose **Restore** on a soft-deleted file to make it active again.

Keep individual files focused on a single topic (for example, one file for account-to-team mappings, another for custom instructions) so the agent can locate the right information efficiently.

Agent memory across sessions

The agent remembers preferences and corrections you provide, then applies them in future sessions. For example, tell the agent that the data platform team owns a set of accounts, that you want costs broken down by Region, or that a class of recommendation does not apply. The agent applies that preference the next time it runs.

Examples of what the agent stores in memory:

- Your name and role.
- AWS account IDs and their owners.
- Preferred cost views or report formats.
- Jira space keys and team assignments.
- Outcomes of past investigations or tasks.
- Corrections you provide during conversation.

You can instruct the agent to remember, update, or forget information through natural language during conversation.

Long conversation handling

The agent operates within a fixed-size context window based on the underlying large language model. Each message in a conversation adds to the context. When the conversation approaches the limit, the agent automatically summarizes older messages to make room for new ones. The agent retains the most recent messages in full and replaces older portions with a summary that preserves the key findings, decisions, and current state of the conversation.

For details on how context files, memory, and reports are stored and secured, see Data protection.

Enable Jira with AWS FinOps Agent

Enable the agent to create issues, add comments, and read issues within the Jira projects you connect. The agent cannot update issue fields (assignee, priority, status transitions), delete issues, or manage Jira workflows. This setup requires Jira Cloud and does not support on-premises Jira.

Integration and connection model

Jira integration uses a two-level model. The **integration** is registered at the AWS account level and stores the OAuth credentials and authorization status for a Jira site. An integration is shared across all agents in the account.

A **connection** is created at the agent level and specifies which Jira project the agent can access. Each connection is bound to one **space key** — the wizard's name for the 2- to 10-letter prefix that Jira uses on issue keys (for example, `ENG` for issues `ENG-1`, `ENG-2`). You grant access only to specific projects, not the full Jira site. Each agent creates its own connections, and the agent can only access projects listed in its connections.

Prerequisites

Before registering a Jira integration, verify that you have the following:

- A Jira Cloud site URL and the space keys of the Jira projects where the agent will create tickets.
- A Jira administrator who can install the AWS FinOps Agent Forge app on the Jira site and authorize the following OAuth permission scopes:

Scope	Description
`read:jira-work`	View Jira issue data.

Scope	Description
`read:jira-user`	View user profiles.
`write:issue:jira`	Create issues. The agent does not use the update permission during preview.
`write:comment:jira`	Create and update comments.
`write:comment.property:jira`	Create and update comment properties.
`write:attachment:jira`	Create and update attachments.
`read:issue:jira`	Read issues. Required by `write:issue:jira` .
`read:comment:jira`	Read comments. Required by `write:comment:jira` .
`read:user:jira`	Read user info.
`read:group:jira`	Read group info.
`read:project:jira`	Read project info.
`read:project-role:jira`	Read project roles.
`read:avatar:jira`	Read avatars.
`read:comment.property:jira`	Read comment properties.
`read:app-system-token`	Enables Forge to pass a token to a remote backend for Atlassian app REST APIs. Required to enable the service.

If your AWS console administrator and Jira administrator are different people, the Jira administrator can configure the integration from the agent detail page within the same account after the agent is created.

Register a Jira integration

Registering a Jira integration is a one-time setup per Jira site. The AWS FinOps Agent console provides a guided wizard with five steps. Before you begin, make sure you are signed in to the Jira Cloud site where you want to install the Forge app in the same browser.

1. **Getting started.** The wizard provides an overview of the integration process and what you will need.
2. **Install Jira Forge app.** Choose **Install** in the wizard. The console opens an Atlassian-hosted install page in a new tab, using your active Jira session. Select the Jira site to install the AWS FinOps Agent app on, and complete the installation in Jira.
3. **Enter installation ID.** After installing the Forge app, navigate to your Jira site's apps section (**Jira Settings > Apps > Manage apps**). The AWS FinOps Agent app appears in the list. Copy the installation ID (a UUID in the format `xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`) and paste it into the wizard.
4. **Authorize and connect.** Choose the authorize button. The wizard redirects you to an Atlassian authorization page. Review the requested permissions and choose **Accept**. After authorization, the wizard creates the integration and advances to the next step.
5. **Complete.** The wizard confirms the integration was created and displays the integration ID and connected Jira site. Choose **Go back to Agents page** to return to the console.

The integration is now registered at the account level and available to any agent in the account.

Create a Jira connection

The integration is registered once at the account level. A connection is created per agent. Because each agent has its own detail page, you create the connection from the detail page of the agent you want to bind to Jira, which specifies the project that agent can access.

1. Open the agent you want to connect from the AWS FinOps Agent console to go to its detail page.
2. Choose **Add connection**, then choose **Jira**. The Jira option is available only after a Jira integration is registered in your account.
3. Select the Jira integration you registered from the dropdown.
4. Enter the **Space key** (2-10 uppercase letters, for example, `ENG` or `DATA`).
5. Choose **Create**.

During preview, each connection binds one space key to the agent. To give the agent access to multiple Jira projects, create a separate connection for each space key.

Capabilities

After a Jira connection is created, the agent has three capabilities within the connected project:

- **Create issues**. When you ask the agent to summarize a cost anomaly or optimization recommendation into a Jira ticket, the agent creates the issue in the connected project.
- **Add comments**. The agent adds follow-up comments to issues it previously created.
- **Read issues**. The agent reads issue metadata (status, assignee, labels) to track whether recommendations have been acted on.

The agent does not select assignees or route tickets to specific engineering teams beyond the connected project. The default assignee comes from your Jira project configuration.

Issue details

Every agent-created issue includes the following:

- A summary with the anomaly or recommendation details.
- A description with affected accounts and services, estimated cost impact, root-cause analysis (for anomalies), implementation guidance (for optimizations), and a link back to the task in the AWS FinOps Agent web application.
- An issue type specified by the agent based on context, defaulting to “Task” when the user does not specify one.

The agent does not set the Assignee or Priority fields. It defers to Jira's project-level default assignee, auto-assignment rules, and priority schemes.

Integration statuses

A Jira integration has four possible statuses:

Status	Meaning
Pending	The administrator has initiated registration but has not completed the OAuth authorization.
Active	OAuth authorization is complete. The agent can use the integration.
Error	The integration was previously active but encountered an issue (expired token, revoked app access, persistent authentication failures). The administrator re-authorizes to restore.
Inactive	The administrator explicitly disabled the integration without deleting it. Configuration and connections are preserved, but the agent does not use it.

Approval workflow

During preview, approval behavior depends on how the action is started. From a chat conversation, the agent presents the action details (issue creation or comment addition) and asks for confirmation before running it. From a scheduled or event-based automation, where you pre-authorized the workflow at setup, the agent runs the action without an additional approval prompt and writes the result to the task history. For details on approval and guardrail behavior, see Agent guardrail controls.

Error handling

When Jira actions fail because of authorization errors (expired OAuth token, revoked app access), the task fails with a descriptive error and the integration status updates to **Error**. The administrator re-authorizes the integration before the agent can resume Jira operations.

When actions fail because of invalid space keys (project deleted or renamed on the Jira site), the agent asks the user which project to use instead for interactive tasks. For scheduled or event-based tasks with no user present, the task fails and the user is notified. If the agent has access to multiple

Jira projects and the user does not specify which one to use, the agent presents the available options and asks the user to choose.

For rate-limited requests (HTTP 429) or site unavailability (HTTP 5xx), the agent retries with exponential backoff up to three times before failing the task.

Enable Slack with AWS FinOps Agent

Enable the agent to post task and automation results to Slack channels you connect. During preview, the connection is one-way: the agent posts to Slack but does not read messages, accept commands, or respond to mentions. The agent cannot edit or delete posts, manage channel membership, or interact with threads after posting.

Integration and connection model

Slack integration uses the same two-level model as Jira. The **integration** is registered at the AWS account level and stores the OAuth credentials and authorization status for a Slack workspace. An integration is shared across all agents in the account.

A **connection** is created at the agent level and specifies which Slack channel the agent posts to. Each agent creates its own connections, and the agent can only post to channels listed in its connections.

Prerequisites

Before registering a Slack integration, verify that you have the following:

- A Slack workspace where you have permission to install apps and authorize third-party OAuth apps.
- The channel IDs of any channels the agent will post to. To find a channel ID in Slack, right-click the channel name, choose **View channel details**, and copy the ID at the bottom (for example, `C04ABCDEF12`).

Register a Slack integration

Registering a Slack integration is a one-time setup per Slack workspace. The AWS FinOps Agent console provides a guided wizard with three steps. Before you begin, make sure you are signed in to the Slack workspace where you want to install the AWS FinOps Agent Slack app in the same browser.

1. **Getting started.** The wizard provides an overview of the integration process and what you will need.
2. **Authorize and connect.** Choose **Authorize with Slack**. The wizard redirects you to Slack to grant permission to the AWS FinOps Agent app. Review the requested permissions and approve. After authorization, Slack returns you to the wizard, which creates the integration automatically.
3. **Complete.** The wizard confirms that Slack is connected and displays the integration ID and the connected workspace name. Choose **Go back to Agents page** to return to the console.

The integration is now registered at the account level and available to any agent in the account.

Create a Slack connection

The integration is registered once at the account level. A connection is created per agent. Because each agent has its own detail page, you create the connection from the detail page of the agent you want to bind to Slack, which specifies the channel that agent can post to.

> **Important**
>
> Add the AWS FinOps Agent Slack app to the channel before you create the connection. From Slack, open the channel, choose the channel name, choose the **Integrations** tab, and add the AWS FinOps Agent app. Without the app in the channel, the agent cannot post messages to this channel.

1. Open the agent you want to connect from the AWS FinOps Agent console to go to its detail page.
2. Choose **Add connection**, then choose **Slack**. The Slack option is available only after a Slack integration is registered in your account.
3. Select the Slack integration you registered from the dropdown.
4. Enter the Slack channel ID. Nine to twelve uppercase alphanumeric characters (for example, `C04ABCDEF12`).
5. Choose **Create**.

Each connection binds one channel to the agent. To give the agent access to multiple Slack channels, create a separate connection for each channel.

Capabilities

After a Slack connection is created, the agent has one capability within the connected channel:

- **Post messages and artifacts**. The agent posts task and automation results, including HTML, PDF, and PPT artifacts, to the connected channel.

Message details

Every agent-posted message includes the following:

- A summary line that identifies the source (task name, automation name, or conversation turn) and a link back to the AWS FinOps Agent web application.
- The result content, formatted in Slack markdown (headings, bold, lists, tables, code blocks).
- Any artifacts the user attached to the message, uploaded as files in the channel. Supported artifact types are `.html`, `.pdf`, and `.pptx`.

Integration statuses

A Slack integration has four possible statuses:

Status	Meaning
Pending	The administrator has initiated registration but has not completed the OAuth authoriza tion with Slack.
Active	OAuth authorization is complete. The agent can post to channels.
Error	The integration was previously active but encountered an issue (revoked OAuth token, the AWS FinOps Agent Slack app removed from the workspace, persistent authentication failures). The administrator re-authorizes to restore.

Status	Meaning
Inactive	The administrator explicitly disabled the integration without deleting it. Configuration and connections are preserved, but the agent does not use it.

Error handling

When Slack actions fail because of authorization errors, the task fails with a descriptive error and the integration status updates to **Error**. The administrator re-authorizes the integration before the agent can resume Slack operations.

When actions fail because the AWS FinOps Agent Slack app is not a member of the connected channel, the post fails. Add the app to the channel from Slack to restore posting.

For rate-limited requests (HTTP 429) or workspace unavailability (HTTP 5xx), the agent retries with exponential backoff up to three times before failing the task.

Monitoring and observability

Logging AWS FinOps Agent API calls using AWS CloudTrail

AWS FinOps Agent is integrated with AWS CloudTrail. CloudTrail provides a record of actions taken by a user, role, or AWS service in AWS FinOps Agent. CloudTrail captures all API calls for AWS FinOps Agent as events, including calls from the AWS FinOps Agent console and code calls to the AWS FinOps Agent API operations.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS FinOps Agent. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**.

Using the information collected by CloudTrail, you can determine which request was sent to AWS FinOps Agent, the source IP address, who made the request, when it was made, and other details. To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AWS FinOps Agent information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS FinOps Agent, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing events with CloudTrail Event history.

For an ongoing record of events in your AWS account, including events for AWS FinOps Agent, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the Amazon S3 bucket that you specify. For more information, see Overview for creating a trail.

CloudTrail records the following AWS FinOps Agent API calls as management events:

- Agent management (`CreateAgentSpace`, `UpdateAgentSpace`, `DeleteAgentSpace`).
- Integration and connection management (`CreateIntegration`, `DeleteIntegration`, `CreateConnection`, `UpdateConnection`, `DeleteConnection`).
- Document management (`CreateDocument`, `UpdateDocument`, `DeleteDocument`, `RestoreDocument`).

- Artifact management (`ListArtifacts`, `GetArtifactContent`, `GetArtifactMetadata`).
- Task and automation management (`CreateTask`, `CancelTask`, `CreateAutomation`, `UpdateAutomation`, `DeleteAutomation`).
- Conversations (`CreateConversation`, `CreateTurn`, `CancelTurn`).
- Agent request responses (`AcceptAgentRequest`, `RejectAgentRequest`).

How caller identity appears in CloudTrail

CloudTrail logs the identity of whoever made the API call:

- **Administrator actions** (`CreateAgentSpace`, `CreateIntegration`, and so on) are logged under the administrator's own IAM identity.
- **Web application actions** (`CreateConversation`, `CreateTask`, `CreateAutomation`, and so on) are logged under the assumed operator role session. The role is shared across web application users, but the calling user's IAM unique identifier is stamped on the session as `sourceIdentity`, so each event in CloudTrail can be attributed back to the individual user who initiated it. This requires the operator role's trust policy to grant `sts:SetSourceIdentity`; see Trust policy.
- **Calls the agent makes to other AWS services** (`ce:GetCostAndUsage`, `cloudtrail:LookupEvents`, and so on) are logged under the assumed agent role session. These appear as standard AWS API activity in CloudTrail and carry the same `sourceIdentity` attribution as web application actions.

To find every action a specific user initiated, filter CloudTrail events by `userIdentity.sessionContext.sourceIdentity` (the user's IAM unique identifier, beginning with `AIDA` for IAM users or `AROA` for assumed roles).

What is not logged in CloudTrail

The agent's internal reasoning, tool calls, and conversation content are not logged to CloudTrail. These are recorded in the agent's internal journal system.

API calls the agent makes to other AWS services (such as Cost Explorer or CloudTrail `LookupEvents`) using the agent's IAM role are logged to CloudTrail under that IAM role's identity, as standard AWS API activity.

Security in AWS FinOps Agent

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive customers.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. AWS FinOps Agent is in preview release and is not yet in scope for AWS compliance programs.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AWS FinOps Agent. The following topics show you how to configure AWS FinOps Agent to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AWS FinOps Agent resources.

Topics

- Data protection in AWS FinOps Agent
- Identity and access management for AWS FinOps Agent
- Agent guardrail controls
- Service improvement
- Amazon Bedrock usage and cross-region inference

Data protection in AWS FinOps Agent

The AWS shared responsibility model applies to data protection in AWS FinOps Agent. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this

infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see Data Privacy FAQ. For information about data protection in Europe, see the General Data Protection Regulation (GDPR) Center.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with AWS FinOps Agent or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Data storage and encryption

Context files, memory files, and artifacts are stored in AWS-managed infrastructure encrypted with AWS KMS. At the start of each session, the agent loads these files into its agent runtime environment. The runtime environment is ephemeral and isolated per session.

Conversation history and task records (including the agent's reasoning traces, tool calls, and results) are stored in AWS-managed infrastructure alongside each conversation and task record.

Data isolation across agents

No content (context files, conversations, memory, artifacts, or task data) is shared across agents. Each agent is a fully isolated instance with its own IAM role, context files, memory, task queue, and artifacts. Information from one agent is not visible or accessible from another.

Managing your data

You control the data AWS FinOps Agent stores on your behalf. The web application supports the actions in the following table during preview. To remove every category at once, delete the agent from the AWS FinOps Agent console.

Data type	How to remove or modify
Context files	Open the **Context files** workspace in the web application. Choose **Delete** to soft-delete a file, which removes it from the agent's active context. Choose **Restore** on a soft-deleted file to make it active again.
Memory entries	Instruct the agent in chat to forget a specific preference, fact, or correction. The agent updates its memory in place.
Conversations	Conversations are retained on the agent and cannot be deleted individually during preview. To remove all conversation history, delete the agent.
Tasks	Cancel a running task from the task detail page. Tasks cannot be deleted individually during preview; task records remain on the agent until you delete the agent.
Automations	Open the **Automations** workspace and delete the automation. Deleting an automation stops new tasks from being created on its

Data type	How to remove or modify
	schedule or trigger; tasks already created by the automation remain on the agent.
Generated artifacts	Open the **Artifacts** workspace and choose **Delete** on the artifact you want to remove.
The entire agent	From the AWS FinOps Agent console Agents page, select the agent and choose **Delete**. Deleting an agent removes the web applicati on link and the previous interaction data the agent stored.

For the AWS Organizations opt-out that prevents your content from being used for service improvement, see Service improvement.

Identity and access management for AWS FinOps Agent

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AWS FinOps Agent resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How AWS FinOps Agent works with IAM
- Identity-based policy examples for AWS FinOps Agent
- Troubleshooting AWS FinOps Agent identity and access

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshooting AWS FinOps Agent identity and access)
- **Service administrator** - determine user access and submit permission requests (see How AWS FinOps Agent works with IAM)
- **IAM administrator** - write policies to manage access (see Identity-based policy examples for AWS FinOps Agent)

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in as a federated identity using credentials from an identity source like AWS IAM Identity Center (IAM Identity Center), single sign-on authentication, or Google/Facebook credentials. For more information about signing in, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

For programmatic access, AWS provides an SDK and CLI to cryptographically sign requests. For more information, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity called the AWS account *root user* that has complete access to all AWS services and resources. We strongly recommend that you don't use the root user for everyday tasks. For tasks that require root user credentials, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users to use federation with an identity provider to access AWS services using temporary credentials.

A *federated identity* is a user from your enterprise directory, web identity provider, or Directory Service that accesses AWS services using credentials from an identity source. Federated identities assume roles that provide temporary credentials.

For centralized access management, we recommend AWS IAM Identity Center. For more information, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy defines permissions when associated with an identity or resource. AWS evaluates these policies when a principal makes a request. Most policies are stored in AWS as JSON documents. For more information about JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Using policies, administrators specify who has access to what by defining which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. An IAM administrator creates IAM policies and adds them to roles, which users can then assume. IAM policies define permissions regardless of the method used to perform the operation.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you attach to an identity (user, group, or role). These policies control what actions identities can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be *inline policies* (embedded directly into a single identity) or *managed policies* (standalone policies attached to multiple identities). To learn how to choose between managed and inline policies, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples include IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. You must specify a principal in a resource-based policy.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Other policy types

AWS supports additional policy types that can set the maximum permissions granted by more common policy types:

- **Permissions boundaries** – Set the maximum permissions that an identity-based policy can grant to an IAM entity. For more information, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – Specify the maximum permissions for an organization or organizational unit in AWS Organizations. For more information, see Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – Set the maximum available permissions for resources in your accounts. For more information, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Advanced policies passed as a parameter when creating a temporary session for a role or federated user. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS FinOps Agent works with IAM

Before you use IAM to manage access to AWS FinOps Agent, learn what IAM features are available to use with AWS FinOps Agent.

IAM feature	AWS FinOps Agent support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys	Yes
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes
Principal permissions	Yes
Service roles	Yes
Service-linked roles	No

To get a high-level view of how AWS FinOps Agent and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for AWS FinOps Agent

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for AWS FinOps Agent

To view examples of AWS FinOps Agent identity-based policies, see Identity-based policy examples for AWS FinOps Agent.

Resource-based policies within AWS FinOps Agent

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for AWS FinOps Agent

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of AWS FinOps Agent actions, see Actions Defined by AWS FinOps Agent in the *Service Authorization Reference*.

Policy actions in AWS FinOps Agent use the following prefix before the action:

```
finops-agent
```

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
      "finops-agent:action1",
      "finops-agent:action2"
         ]
```

To view examples of AWS FinOps Agent identity-based policies, see Identity-based policy examples for AWS FinOps Agent.

Policy resources for AWS FinOps Agent

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of AWS FinOps Agent resource types and their ARNs, see Resources Defined by AWS FinOps Agent in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS FinOps Agent .

To view examples of AWS FinOps Agent identity-based policies, see Identity-based policy examples for AWS FinOps Agent.

Policy condition keys for AWS FinOps Agent

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match

the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of AWS FinOps Agent condition keys, see Condition Keys for AWS FinOps Agent in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions Defined by AWS FinOps Agent .

To view examples of AWS FinOps Agent identity-based policies, see Identity-based policy examples for AWS FinOps Agent.

ACLs in AWS FinOps Agent

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with AWS FinOps Agent

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with AWS FinOps Agent

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Cross-service principal permissions for AWS FinOps Agent

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for AWS FinOps Agent

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break AWS FinOps Agent functionality. Edit service roles only when AWS FinOps Agent provides guidance to do so.

Service-linked roles for AWS FinOps Agent

Supports service-linked roles: No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for AWS FinOps Agent

By default, users and roles don't have permission to create or modify AWS FinOps Agent resources. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Create IAM policies (console) in the *IAM User Guide*.

For details about actions and resource types defined by AWS FinOps Agent, including the format of the ARNs for each of the resource types, see Actions, Resources, and Condition Keys for AWS FinOps Agent in the *Service Authorization Reference*.

Topics

- Policy best practices
- Using the AWS FinOps Agent console
- Allow users to view their own permissions

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS FinOps Agent resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to

specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Using the AWS FinOps Agent console

To access the AWS FinOps Agent console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS FinOps Agent resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can use the AWS FinOps Agent console, attach an IAM identity-based policy that grants the actions needed for the workflows the user performs. The required actions depend on the agent being created or managed; for details, see Actions Defined by AWS FinOps Agent in the *Service Authorization Reference*. For more information, see Adding permissions to a user in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Troubleshooting AWS FinOps Agent identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AWS FinOps Agent and IAM.

Topics

- I am not authorized to perform an action in AWS FinOps Agent
- I am not authorized to perform iam:PassRole
- I want to allow people outside of my AWS account to access my AWS FinOps Agent resources

I am not authorized to perform an action in AWS FinOps Agent

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `finops-agent:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform: finops-
agent:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `finops-agent:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to AWS FinOps Agent.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AWS FinOps Agent. However, the action requires the service to have

permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my AWS FinOps Agent resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AWS FinOps Agent supports these features, see How AWS FinOps Agent works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Agent guardrail controls

Read-only actions

Read-only actions run autonomously without requiring approval. These include querying cost data, retrieving optimization recommendations, searching memory, and reading context files.

Read-write actions

Read-write actions affect systems outside the agent. AWS FinOps Agent supports three read-write actions: creating a Jira issue, adding a Jira comment, and posting a Slack message. The agent's approval behavior depends on the action and on how it is started.

Jira issue creation and comment addition

Jira issue creation and comment addition are the read-write actions that require approval, and they require approval only when started from a chat conversation or from an on-demand task. The agent presents the action details (for an issue: project, summary, description; for a comment: target issue and body), and you confirm or revise before the action runs.

When the action is part of a scheduled or event-based automation, no approval is required. You pre-authorize the agent to create Jira issues and add comments when you set up the automation, so the workflow runs end to end without further prompting. You can review every action the agent took in the task history.

Slack messages

Posting messages to Slack channels does not require approval, including when the post is started from a chat conversation or an on-demand task. The agent posts directly to the channel you specify and writes the result to the task history.

Behavioral guardrails

AWS FinOps Agent is constrained to answer questions within the FinOps domain. The agent focuses on AWS cost management and optimization topics, does not generate answers outside the FinOps domain, and responds in English by default. These constraints are not administrator-configurable.

Generative AI accuracy considerations

AWS FinOps Agent is a generative AI service built on Amazon Bedrock. You are responsible for all decisions made, advice given, actions taken, and failures to take action based on your use of the service. Output generated by the underlying large language model is probabilistic. Evaluate it for accuracy as appropriate for your use case, including by employing human review of such output. You can review the agent's reasoning steps and tool calls in the web application chat area.

Known accuracy considerations:

- Cost data accuracy depends on the underlying AWS APIs. The agent retrieves data from Cost Explorer, Cost Anomaly Detection, and other billing services and does not modify the raw data. Discrepancies between the agent's responses and console views might occur because of differences in time ranges, granularity, or filtering.
- Root-cause analysis for cost anomalies is investigative. The agent correlates cost trends from Cost Explorer with CloudTrail events during anomaly investigations to identify likely causes, but the identified root cause might not always be correct. Treat root-cause analysis as a starting point for investigation rather than a definitive conclusion.
- Report generation creates charts and visualizations from cost data. Verify that charts accurately represent the underlying data, particularly for complex multi-dimensional breakdowns.
- Optimization recommendations come from Cost Optimization Hub and Compute Optimizer. The agent summarizes and contextualizes these recommendations but does not generate its own optimization analysis.

Service improvement

Content used for service improvement

AWS may use certain content from AWS FinOps Agent for AWS FinOps Agent service improvement, for example to provide better responses to common questions, fix operational issues, or debug. Content that AWS may use for service improvement includes your prompts, the agent's responses, task execution data, tool call patterns, usage telemetry, and files you upload as context.

How to opt out

You can opt out of having your content used to develop or improve the quality of AWS FinOps Agent (and other covered AWS AI services) by configuring an AWS Organizations AI services opt-out policy. The opt-out policy applies to all current and future supported AWS AI services. To opt out, see AI services opt-out policies in the *AWS Organizations User Guide*.

Amazon Bedrock usage and cross-region inference

During preview, AWS FinOps Agent runs in the US East (N. Virginia) Region (`us-east-1`). Your data, including context files, conversations, memory, and artifacts, remains stored in `us-east-1`.

AWS FinOps Agent uses Amazon Bedrock cross-region inference to improve performance and reliability. Cross-region inference requests are kept within the geography where the request

originated. For example, requests from AWS Regions in the United States stay within AWS Regions in the United States. Agent inference may be processed outside the specific Region but remains within the same geography. All data is encrypted in transit. Cross-region inference does not change where your data is stored.

Quotas

The following table describes the quotas for AWS FinOps Agent.

Name	Default	Adjustable	Description
Agents per account per Region	1	Yes	The default number of agents that you can create per account in each AWS Region.
Artifact storage per agent	100 MB	Yes	The default amount of storage for artifacts (such as cost reports and investigation outputs) that the agent retains per agent.
Context file size per upload	10 MB	No	The maximum size of a single context file that you can upload to an agent.
Context file storage per agent	100 MB	Yes	The default total storage for all context files uploaded to an agent.
Jira integrations per account	1	No	The maximum number of Jira integrations (account-level connections to a Jira Cloud site) that you can register per account.
Slack integrations per account	1	No	The maximum number of Slack integrations (account-level connections to a Slack workspace) that you can register per account.
Jira connections per agent	2	No	The maximum number of Jira connections (agent-level bindings to a Jira project) that you can create per agent.
Slack connections per agent	2	No	The maximum number of Slack connections (agent-level bindings to a

Name	Default	Adjustable	Description
			Slack channel) that you can create per agent.

Requesting a quota increase

During preview, request a quota increase by opening an AWS Support case:

1. Open the Create case page in the AWS Support Center.
2. For **Service**, choose **AWS FinOps Agent**.
3. For **Category**, choose **General**.
4. Provide the quota name, your account ID, agent name and agent ID, the requested new value, and a brief justification.
5. Submit the case.

Document history for AWS FinOps Agent

The following table describes the documentation releases for AWS FinOps Agent.

Change	Description	Date
Public preview	This is the initial preview release of AWS FinOps Agent (preview).	June 9, 2026