



User Guide

AWS Startup Advisor



AWS Startup Advisor: User Guide

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

What is AWS Startup Advisor?	1
Features of AWS Startup Advisor	1
How AWS Startup Advisor uses your startup context	2
Are you a first-time AWS Startup Advisor user?	2
Accessing AWS Startup Advisor	2
AWS Startup Advisor pricing	3
Setting up AWS Startup Advisor	4
Prerequisites	4
Connect your AWS profile	4
Getting started	5
Prerequisites	5
Get started on the web	5
Install the Claude Code plugin	5
Install the IDE extension	6
Install the extension for Cursor	6
Install the extension for VS Code	7
Install the extension for Kiro	7
Sign in	7
Install the AWS Startup Advisor skills in other coding agents (npx)	8
Next steps	8
MCP server	10
Available MCP server	10
Provision the MCP server	10
Using skills	12
Start and grow on AWS	12
start-building-for-startups	12
architect-for-startups	12
knowledge-base-for-startups	13
prompt-library-for-startups	13
contextual-offers-for-startups	13
Build AI agents on AWS	13
agent-advisor	13
Migrate to AWS	14
gcp-to-aws	14

azure-to-aws	14
heroku-to-aws	14
llm-to-bedrock	14
tf-best-practices	15
Using the IDE extension	16
Alerts	16
What the Alerts tab monitors	16
Refresh alerts and switch profiles or Regions	17
Configure alert settings	17
Credits and cost widget	17
Prompts	18
Use a prompt	18
Prompt categories	18
Security	20
Data protection	21
Data and privacy in AWS Startups	21
Encryption at rest	22
Encryption in transit	22
Identity and access management	22
Audience	23
Authenticating with identities	23
Managing access using policies	27
How AWS Startups works with IAM	29
AWS Startups identity-based policy examples	31
Troubleshooting AWS Startups identity and access	34
Compliance validation	35
Resilience	36
Infrastructure security	36
CloudTrail logs	37
AWS Startups information in CloudTrail	37
Understanding AWS Startups log file entries	38
VPC endpoints (AWS PrivateLink)	41
Considerations for AWS Startups VPC endpoints	41
Creating an interface VPC endpoint for AWS Startups	41
Creating a VPC endpoint policy for AWS Startups	42

What is AWS Startup Advisor?

AWS Startup Advisor is an AI-powered build assistant for startups. It provides AWS-vetted, Solutions Architect-curated skills and guidance for startups on account setup, architecture, Activate credits, cost, security, migration, AI, and scalability. AWS Startup Advisor delivers this guidance as skills that founders and their coding agents can use. You receive recommendations directly in your development environment, so you can act on them as you build.

This guide is for startup founders and startup developers who build on AWS. You can use AWS Startup Advisor at any stage, whether you are building your product from scratch, have an existing product or service you are enhancing or scaling, or if you are migrating from another cloud provider to AWS.

Topics

- [Features of AWS Startup Advisor](#)
- [How AWS Startup Advisor uses your startup context](#)
- [Are you a first-time AWS Startup Advisor user?](#)
- [Accessing AWS Startup Advisor](#)
- [AWS Startup Advisor pricing](#)

Features of AWS Startup Advisor

AWS Startup Advisor provides a skills-based experience. Each skill helps you complete a common startup task with AWS-vetted guidance. The following skills are available:

- **Set up your AWS environment** – Configure IAM, billing alerts, and security baselines.
- **Scaffold your infrastructure** – Get AWS-vetted infrastructure recommendations in your editor.
- **Deploy local code to AWS** – Move local code to AWS, from architecture to permissions.
- **Optimize costs** – Get specific cost-saving actions for your workloads.
- **Harden security for your stage** – Apply security guidance that fits your stage, whether you build an MVP or sell to enterprises.
- **Migrate to AWS** – Move your workloads to AWS from another provider.
- **Deploy an AI agent** – Deploy an AI agent to AWS.

For more information about each skill, see [Using AWS Startup Advisor skills](#).

How AWS Startup Advisor uses your startup context

AWS Startup Advisor can pick up where your AWS Activate credit application leaves off. Instead of starting from scratch, if you applied for AWS Activate credits, you start with context about your startup. AWS Startup Advisor uses two sources to understand your tech stack and your startup's stage:

- The information in your AWS Activate credit application
- Your startup's website

AWS Startup Advisor combines this context with expertise from AWS Startup Solutions Architects. As a result, the guidance that you receive is tailored to your startup's stack and stage.

AWS Startup Advisor also has context about your tech stack from your project. When you work with your AI coding agent in your source code and it invokes an AWS Startup Advisor skill, the skill can use your project to understand your actual tech stack, so the guidance reflects what you are building.

Are you a first-time AWS Startup Advisor user?

If you are a first-time user of AWS Startup Advisor, we recommend that you begin by reading the following sections:

- [Setting up AWS Startup Advisor](#)
- [Getting started with AWS Startup Advisor](#)
- [Using AWS Startup Advisor skills](#)

Accessing AWS Startup Advisor

You can access AWS Startup Advisor in any of the following ways. You get the same skills and the same context across all of them.

- **Web** – Get started on the web at <https://startups.aws.com/>.
- **IDE extension** – Install the AWS Startup Advisor IDE extension from the marketplace for your IDE. For VS Code, download and install it from the [Visual Studio Marketplace](#). For Cursor and

Kiro, download and install it from the [Open VSX Registry](#). The IDE extension works inside Kiro or any VS Code-based IDE.

- **Claude Code plugin** – Install the plugin from [the AWS Startup Advisor plugin page](#). The plugin works within Claude Code.

AWS Startup Advisor pricing

AWS Startup Advisor is available at no additional charge. You pay only for the AWS resources you or your agent provisions or interacts with, at standard AWS pricing.

Setting up AWS Startup Advisor

Before you use AWS Startup Advisor, review the following. You can use AWS Startup Advisor without an AWS account, but an AWS account is recommended, and it is required to deploy and build on AWS.

Topics

- [Prerequisites](#)
- [Connect your AWS profile](#)

Prerequisites

You can start using AWS Startup Advisor without an AWS account or an AWS Activate application. The following help you get the most from AWS Startup Advisor:

- **An AWS account (recommended)** – An AWS account is not required to use AWS Startup Advisor, but we recommend one, and it is required to deploy and build on AWS. If you do not have an AWS account, create one at [the AWS sign-up page](#).
- **AWS Activate credits (optional)** – Submitting an AWS Activate Credit application is not required. If you are eligible and want to take advantage of AWS Activate credits, you are welcome to apply. When you have a submitted application, AWS Startup Advisor can use it to seed context about your startup and tailor its guidance. For more information, see [AWS Activate](#).

Connect your AWS profile

AWS Startup Advisor uses your AWS profile permissions to work with your configured resources.

To use the IDE extension, you sign in with an AWS CLI profile (from `~/.aws/credentials` or `~/.aws/config`), IAM Identity Center (SSO), or IAM access keys. A read-only profile is sufficient for alerts. For the sign-in steps, see [Sign in](#) in Getting started. The extension also includes **Get started** prompts that can help you set up read-only access or IAM Identity Center (SSO) access for the extension. For more information, see [Prompts](#).

The AWS credits and cost tracker widget in the AWS Startup Advisor IDE extension requires AWS billing and cost read permissions. For more information, see [Identity and access management for AWS Startups](#).

Getting started with AWS Startup Advisor

You can get started with AWS Startup Advisor on the web, in a supported IDE, or in Claude Code. Choose the entry point that fits your workflow. The same curated skills are available across all access methods, so you get the same guidance no matter which one you choose.

Topics

- [Prerequisites](#)
- [Get started on the web](#)
- [Install the Claude Code plugin](#)
- [Install the IDE extension](#)
- [Install the AWS Startup Advisor skills in other coding agents \(npx\)](#)
- [Next steps](#)

Prerequisites

Before you begin, review the prerequisites in [Setting up AWS Startup Advisor](#). An AWS account is recommended, and you need one to deploy and build on AWS. A submitted AWS Activate Credit application is optional. If you submit one, AWS Startup Advisor can use it to tailor its guidance to your startup. The same curated skills are available across all access methods.

Get started on the web

You can start using AWS Startup Advisor on the web. No install is required.

1. Open <https://startups.aws.com/>.
2. Answer a few questions about your startup.

AWS Startup Advisor uses your AWS Activate Credit application to pick up your stack and stage.

Install the Claude Code plugin

You can install the AWS Startup Advisor plugin for Claude Code. To install the plugin, run the following command in your terminal:

```
claude plugin install aws-startup-advisor@claude-plugins-official
```

The plugin works on macOS, Windows, and Linux. For the plugin's marketplace listing, see [the AWS Startup Advisor Claude Code Plugin Marketplace page](#). To download Claude Code, see [the Claude Code page](#).

Install the IDE extension

You can install the AWS Startup Advisor extension for Cursor, VS Code, or Kiro. Other VS Code-based IDEs can install the extension from the [Open VSX Registry](#).

Install the tool you want to use before you install the extension. For more information, see the official download site for each tool.

The simplest way to install is from within your IDE. Search for **AWS Startup Advisor** in your IDE's Extensions view and choose **Install**. The per-IDE command-line and marketplace options follow.

Note

Requirements: VS Code 1.85 or later (or Kiro or Cursor), an AWS account, and an AWS CLI profile, an IAM Identity Center (SSO) start URL, or IAM access keys. A read-only profile is sufficient, because the extension never writes to your account.

Install the extension for Cursor

To download Cursor, see [the Cursor downloads page](#).

The Cursor extension is available from the Open VSX Registry. For the marketplace listing, see [AWS Startup Advisor on the Open VSX Registry](#).

To install the extension from the command line, run the following command:

```
cursor --install-extension amazonwebservices.aws-startup-advisor
```

This command requires the `cursor` CLI command on your PATH. To add it in Cursor, open the Command Palette and choose **Install 'cursor' command in PATH**.

Alternatively, you can install the extension from within Cursor:

1. Open Extensions (**Cmd+Shift+X** on macOS, or **Ctrl+Shift+X** on Windows and Linux).
2. Search for **AWS Startup Advisor**.
3. Choose **Install**.

Install the extension for VS Code

To download VS Code, see [the VS Code downloads page](#).

The VS Code extension is available from the Visual Studio Marketplace. For the marketplace listing, see [AWS Startup Advisor on the Visual Studio Marketplace](#).

To install the extension from the command line, run the following command:

```
code --install-extension AmazonWebServices.aws-startup-advisor
```

This command requires the code CLI command on your PATH. To add it in VS Code, open the Command Palette and choose **Shell Command: Install 'code' command in PATH**.

Alternatively, you can install the extension from the Visual Studio Marketplace listing by choosing **Install**, which uses the `vscode:extension/AmazonWebServices.aws-startup-advisor` deep link.

Install the extension for Kiro

To download Kiro, see [the Kiro downloads page](#).

The Kiro extension is available from the Open VSX Registry. For the marketplace listing, see [AWS Startup Advisor on the Open VSX Registry](#).

To install the extension from the command line, run the following command:

```
kiro --install-extension amazonwebservices.aws-startup-advisor
```

This command requires the kiro CLI command on your PATH. To add it in Kiro, open the Command Palette and choose **Install 'kiro' command in PATH**.

Sign in

After you install the extension, sign in to connect it to your AWS account. Signing in enables the extension's alerts, because you give the extension permission to read your account. You can sign in

using one of the following methods: AWS IAM Identity Center (SSO) (recommended), an AWS CLI profile, or IAM access keys.

1. Choose the AWS Startup Advisor icon in the activity bar.
2. Choose one of the following sign-in options:
 - **Sign in with SSO** – Paste your IAM Identity Center start URL and follow the browser flow.
 - **Sign in with AWS profile** – Choose a profile.
 - **Sign in with IAM access keys** – Enter your IAM access key ID and secret access key.

After you sign in, the extension takes a few seconds to read your account. The sidebar then populates with alerts.

Install the AWS Startup Advisor skills in other coding agents (npx)

For agents beyond the IDE extension and Claude Code, for example Codex or GitHub Copilot, you can install just the skills with the skills CLI. This path requires Node.js. To download Node.js, see [the Node.js downloads page](#).

To install the skills, run the following command:

```
npx skills add https://github.com/aws/agent-toolkit-for-aws/tree/main/plugins/aws-startup-advisor --skill '*'
```

If you get an npm authentication or registry error, retry against the public registry:

```
npx --registry https://registry.npmjs.org skills add https://github.com/aws/agent-toolkit-for-aws/tree/main/plugins/aws-startup-advisor --skill '*'
```

Installing the skills requires no AWS credentials. This path installs the skills, but it does not configure MCP servers. For more information, see [Configure live data with MCP servers \(optional\)](#).

Next steps

How you get the AWS Startup Advisor skills depends on how you installed AWS Startup Advisor:

- **Claude Code plugin** – The skills are included when you install the plugin. You can start using them right away.
- **IDE extension (VS Code, Cursor, or Kiro)** – Installing the extension provides alerts and prompts, but not the skills. To add the skills, choose **Install AWS Startup Advisor skills** in the banner that the extension displays. After the skills install, they are available in your coding agent, which invokes them when they are relevant to your request.
- **Other coding agents (npx)** – The `npx skills` add command installs the skills directly, as described in the previous section.

For more information, see [Using AWS Startup Advisor skills](#).

Configure live data with the MCP server (optional)

AWS Startup Advisor can use a Model Context Protocol (MCP) server for live AWS data. The MCP server is an optional enhancement, not a requirement. Skills run without it, but they cannot fetch live AWS data, such as documentation, regional service availability, and live API calls. The MCP server is independent of skills, and it does not serve the skills that you install locally.

Topics

- [Available MCP server](#)
- [Provision the MCP server](#)

Available MCP server

AWS Startup Advisor can use the following MCP server:

- `aws-mcp` – The AWS MCP Server. This stdio server launches through `uvx`. It provides access to current AWS documentation (search and read), regional service availability, authenticated AWS API calls, sandboxed Python script execution, and on-demand skill retrieval and recommendations. For more information, see [Understanding AWS MCP Server tools](#).

This server runs through `uvx`, which requires `uv` and `uvx` on your machine. To install `uv`, see [the uv documentation](#). The server launches with the following command.

```
uvx mcp-proxy-for-aws-cli@latest https://aws-mcp.us-east-1.api.aws/mcp
```

Provision the MCP server

When you install the Claude Code plugin from the Claude Code Plugin Marketplace, the `aws-mcp` server is configured automatically because it is bundled with the plugin. For more information, see [Install the Claude Code plugin](#).

The `npx skills add path` installs only the skills, not the MCP server. To use the MCP server with another agent, such as Kiro, Cursor, Codex, or fx, add the AWS MCP Server to that agent's own MCP configuration. Then, install the skills separately. For example, add the server to the Kiro `.kiro/settings/mcp.json` file or the Cursor `.cursor/mcp.json` file with the following command.

```
uvx mcp-proxy-for-aws-cli@latest https://aws-mcp.us-east-1.api.aws/mcp
```

Using AWS Startup Advisor skills

AWS Startup Advisor bundles 11 skills that a coding agent uses. The same skills are available across the supported agents, so you get the same guidance no matter which agent you use. The skills fall into three groups: skills that help you start and grow on AWS, skills that build AI agents on AWS, and skills that migrate workloads to AWS.

To use a skill, prompt your coding agent with what you want to do. Each skill in the following sections includes an example prompt.

Topics

- [Start and grow on AWS](#)
- [Build AI agents on AWS](#)
- [Migrate to AWS](#)

Start and grow on AWS

This group scaffolds new projects, tailors architecture guidance to your stage, and provides reference material and prompts.

start-building-for-startups

The `start-building-for-startups` skill is an interactive discovery workflow. It scans your codebase, asks about your goals and constraints through picker questions, and then writes an AWS architectural scaffold directly into your project.

Example prompt: "Scaffold a serverless API for my startup."

architect-for-startups

The `architect-for-startups` skill provides stage-aware AWS architecture guidance. It adjusts recommendations based on your startup stage (pre-revenue, seed, Series A, or Series B and later), team size, runway, and timeline.

Example prompt: "Recommend an architecture for my seed-stage SaaS."

knowledge-base-for-startups

The `knowledge-base-for-startups` skill is an AWS Startups knowledge base. It includes the AWS Activate FAQ, a credits guide, programs, partner offers, sample architectures, and learn articles. It is a read-only reference that is searchable and available offline after install.

Example prompt: "Do my AWS Activate credits expire?"

prompt-library-for-startups

The `prompt-library-for-startups` skill provides about 30 AWS-curated, copy-paste prompts, for example MVP scaffolding, a RAG chatbot, a security baseline, cost anomaly detection, EKS deployment, and a Well-Architected review. It also provides downloadable installable agents: Migration, Multi-Account Transition Advisor, Bill Shock Preventer, and Service Quota.

Example prompt: "Give me a prompt for an MVP on AWS."

contextual-offers-for-startups

The `contextual-offers-for-startups` skill is an offer layer that runs after another skill produces a recommendation, plan, or build. When a genuinely relevant AWS Activate partner offer exists for what you're building or migrating, it appends one quiet, dismissible line with a redeem link. It never changes the technical advice, and it adds at most one offer for each response.

Build AI agents on AWS

This group helps you choose a runtime and build AI agents on AWS.

agent-advisor

The `agent-advisor` skill helps you pick an AWS runtime for AI agents, such as Amazon Bedrock AgentCore, ECS, EKS, Lambda, or Lambda MicroVMs. It can generate a migration plan for existing agent workloads and build a deployable proof of concept, and it also covers Temporal workers. Runtime recommendations and design-backed proofs of concept work standalone. The full migration-plan step additionally needs the `gcp-to-aws` skill installed, and it degrades gracefully without it: the step is skipped, not broken.

Example prompt: "Which AWS runtime should I use for my AI agent?"

Migrate to AWS

The migration skills plan and run moves from other cloud providers and AI providers to AWS.

gcp-to-aws

The `gcp-to-aws` skill plans a migration from Google Cloud, and from AI providers such as OpenAI, Gemini, and LangChain, to AWS through a guided multi-phase workflow. It discovers resources from Terraform and other infrastructure as code, application code, and Google Cloud billing exports. It then designs an AWS architecture, estimates cost, and generates migration artifacts, including Terraform.

Example prompt: "Help me migrate from GCP to AWS."

azure-to-aws

The `azure-to-aws` skill plans a migration from Microsoft Azure to AWS through a guided six-phase workflow: discover, clarify, design, estimate, generate artifacts, and gather feedback. It discovers resources from Terraform (`azurearm_*`), a read-only, consent-gated live capture from the Azure CLI (`az`), application code, and billing exports. Bicep and Azure Resource Manager (ARM) templates aren't supported yet. After the estimate, you can run an optional what-if repricing workshop, and generating artifacts is opt-in.

Example prompt: "Help me migrate from Azure to AWS."

heroku-to-aws

The `heroku-to-aws` skill plans a Heroku-to-AWS migration. By default it maps Dynos to Elastic Beanstalk, with Fargate or EKS overrides available. It maps Postgres to RDS or Aurora, Redis to ElastiCache, and Kafka to MSK, and it handles common add-ons. After the estimate, you can run an optional what-if repricing workshop.

Example prompt: "Plan a migration of my Heroku app to AWS."

llm-to-bedrock

The `llm-to-bedrock` skill runs an OpenAI, Gemini, or Anthropic to Amazon Bedrock SDK rewrite. It assesses the codebase, rewrites call sites, evaluates output quality against Amazon Bedrock, and delivers a ready-to-merge git branch. This skill requires the `gcp-to-aws` skill installed alongside it, because it delegates the assessment step to that skill.

Example prompt: "Move my OpenAI app to Amazon Bedrock."

tf-best-practices

The `tf-best-practices` skill provides best-practice authoring guidance and a read-only policy gate for the AWS Terraform that the migration skills generate.

Example prompt: "Review my generated Terraform against AWS best practices."

Note

The skills are cross-aware. They consult each other and route migration and agent intent to the matching skill.

Using the AWS Startup Advisor IDE extension

The AWS Startup Advisor IDE extension (for VS Code, Kiro, and Cursor) has three main capabilities: personalized **alerts** about your AWS account, a **credits and cost widget** for tracking your startup's AWS credit balance and AWS costs, and a curated library of **prompts** that you can use with your AI coding agent. To use the extension, you sign in to your AWS account (see the Sign in steps in [Getting started](#)). When you sign in, you give the extension permission to read your account and generate alerts. You can sign in using one of the following methods: AWS IAM Identity Center (SSO) or an AWS CLI profile. You can switch between multiple profiles or accounts and Regions.

Topics

- [Alerts](#)
- [Credits and cost widget](#)
- [Prompts](#)

Alerts

The Alerts tab in the AWS Startup Advisor IDE extension surfaces optimization opportunities and recommended actions. AWS Startup Advisor uses your AWS profile permissions to monitor your configured resources. It proactively flags what to fix, so you can act on issues as they arise.

What the Alerts tab monitors

The AWS Startup Advisor IDE extension surfaces personalized alerts. It generates most alerts locally from read-only AWS API calls against your account, scoped per Region. The alerts fall into three categories:

- **Security** – For example, root access keys, missing MFA, unused credentials, open security groups, default VPCs, unencrypted RDS, EBS, or ElastiCache resources, CloudTrail gaps, and API Gateway throttling.
- **Scalability** – For example, missing CloudWatch alarms and throttled Lambda functions.
- **Cost management** – For example, stopped EC2 instances, orphaned EBS volumes, unattached Elastic IP addresses, orphaned RDS snapshots, idle SageMaker notebooks, Compute Optimizer recommendations, and AWS credits and balances.

Note

The AWS credits and cost information comes from the credits and cost widget, which works differently from the locally computed alerts. For details, see [Credits and cost widget](#).

Refresh alerts and switch profiles or Regions

You can refresh alerts on demand. Alerts are scoped per Region.

- To refresh alerts, run the **AWS Startup Advisor: Refresh Alerts** command.
- To switch your AWS profile or Region, use the avatar menu. Alerts are scoped per Region, so switching the Region changes which alerts appear.

Configure alert settings

You can configure the following settings for the AWS Startup Advisor extension:

- `awsStartupAdvisor.alertPollInterval` – How often, in milliseconds, the extension polls for alerts. The default is 15000.
- `awsStartupAdvisor.telemetryEnabled` – Whether telemetry is enabled.
- `awsStartupAdvisor.logLevel` – The log verbosity level.

Credits and cost widget

The AWS Startup Advisor IDE extension includes a credits and cost widget that tracks your startup's AWS credit balance and your AWS costs for your AWS account. To use it, sign in to your AWS account in the extension (see the Sign in steps in [Getting started](#)).

Unlike the alerts, which the extension computes locally from read-only AWS API calls, the credits and cost widget calls the AWS Startups service operation `startups:GetSpendSummary` to retrieve your account's credits and cost data. The AWS Startups service obtains this data from AWS Billing and Cost Management and AWS Cost Explorer on your behalf. Because of this, the credits and cost widget requires AWS billing and cost read permissions. For more information, see [Identity and access management for AWS Startups](#).

Prompts

The extension includes a curated library of prompts vetted by AWS Startup Solutions Architects, grouped by category. These prompts are a curated selection drawn from the broader AWS Startups prompt library. They are distinct from the `prompt-library-for-startups` skill, which is the fuller prompt-and-agent library used by coding agents. For more information about that skill, see [Using AWS Startup Advisor skills](#).

Use a prompt

In the extension sidebar, browse prompts by category, open a prompt to review its steps, and then choose **Copy prompt**. Paste it into your AI coding agent (for example, Amazon Q, Kiro Chat, Cursor Chat, GitHub Copilot, or Claude Code) and run it.

Prompt categories

The prompts are grouped into the following categories. The following lists show representative examples, not every available prompt.

- **Get started** – For example, set up read-only access for the extension, and set up IAM Identity Center (SSO) access for the extension.
- **Build** – For example, Day 1 AWS account setup, scaffold your AWS architecture, deploy local code to AWS, and build AI workloads on AWS.
- **Migrate** – For example, migrate from GCP to AWS, migrate from Heroku to AWS, and migrate AI workloads to AWS.
- **Cost** – For example, cost and credits, and Compute Optimizer.
- **Security** – For example, security baseline assessment, and automated GuardDuty and Security Hub deployment.
- **Scalability** – For example, Well-Architected review, resiliency baseline, and requesting more GPU or Amazon Bedrock quota.

Note

The prompts are starting points that you run in your agent. To have your agent carry out the deeper multi-step workflows (for example, the migrations), install the AWS Startup Advisor skills. You will see a notification when you open the AWS Startup Advisor IDE

extension for the first time prompting you to install the AWS Startup Advisor skills. For more information, see [Using AWS Startup Advisor skills](#) and, for live data, [Configure live data with MCP servers \(optional\)](#). The extension also surfaces a banner that prompts you to install the skills.

Security in AWS Startups

AWS Startup Advisor uses the AWS Startups service. The following topics describe the security of the AWS Startups service.

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

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](#). To learn about the compliance programs that apply to AWS Startups, see [AWS Services in Scope by Compliance Program](#).
- **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 Startups. It shows you how to configure AWS Startups 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 Startups resources.

Contents

- [Data protection in AWS Startups](#)
- [Identity and access management for AWS Startups](#)
- [Compliance validation for AWS Startups](#)
- [Resilience in AWS Startups](#)
- [Infrastructure security in AWS Startups](#)
- [Logging AWS Startups API calls using AWS CloudTrail](#)

Data protection in AWS Startups

The AWS [shared responsibility model](#) applies to data protection in AWS Startups. 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. This content includes the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the [Data Privacy FAQ](#). For information about data protection in Europe, see the [AWS Shared Responsibility Model and GDPR](#) blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with 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 recommend TLS 1.2 or later.
- Set up API and user activity logging with AWS CloudTrail.
- 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 140-2 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 \(FIPS\) 140-2](#).

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with AWS Startups or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into AWS Startups or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

Data and privacy in AWS Startups

AWS Startups is designed to protect your data and your privacy. AWS Startups uses only the following two sources of data to tailor its guidance to your startup:

- The information in your AWS Activate credit application
- Your startup's website

AWS Startups uses this data solely to tailor the experience to your stack and your stage. AWS Startups protects your data in the following ways:

- **Your source code is not stored.** AWS Startups does not store your source code.
- **Your data is not shared with third parties.** AWS Startups does not share your data with third parties.
- **Your data is used only to tailor your experience.** AWS Startups uses your data solely to tailor its guidance to your startup.

Encryption at rest

AWS Startups does not store customer data at rest. Because the service does not store or retain your data, encryption at rest does not apply.

Encryption in transit

AWS Startups encrypts all data in transit by using TLS 1.2 or later. This applies both to traffic between you and the service and to traffic between the service and downstream AWS services, such as AWS Cost Explorer and AWS Billing and Cost Management.

For the AWS credits and cost tracker, AWS Startups retrieves your billing and cost data from AWS Cost Explorer and AWS Billing and Cost Management in real time on your behalf. AWS Startups does not store or retain this data.

Identity and access management for AWS Startups

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 Startups 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 Startups works with IAM](#)
- [AWS Startups identity-based policy examples](#)
- [Troubleshooting AWS Startups identity and access](#)

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work you do in AWS Startups.

Service user – If you use the AWS Startups service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS Startups features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AWS Startups, see [Troubleshooting AWS Startups identity and access](#).

Service administrator – If you're in charge of AWS Startups resources at your company, you probably have full access to AWS Startups. It's your job to determine which AWS Startups features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AWS Startups, see [How AWS Startups works with IAM](#).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AWS Startups. To view example AWS Startups identity-based policies that you can use in IAM, see [AWS Startups identity-based policy examples](#).

Authenticating with identities

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

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see [How to sign in to your AWS account](#) in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see [Signature Version 4 signing process](#) in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see [Multi-factor authentication](#) in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide* and [Using multi-factor authentication \(MFA\) in AWS](#) in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see [Tasks that require root user credentials](#) in the *Account Management Reference Guide*.

Federated identity

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

A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For

information about IAM Identity Center, see [What is IAM Identity Center?](#) in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

IAM users and groups

An [IAM user](#) is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see [Rotate access keys regularly for use cases that require long-term credentials](#) in the *IAM User Guide*.

An [IAM group](#) is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see [When to create an IAM user \(instead of a role\)](#) in the *IAM User Guide*.

IAM roles

An [IAM role](#) is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by [switching roles](#). You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see [Using IAM roles](#) in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see [Creating a role for a third-party Identity Provider](#) in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permission sets, see [Permission sets](#) in the *AWS IAM Identity Center (successor to AWS Single Sign-On) User Guide*.

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
 - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions.
 - **Service role** – 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 [Creating a role to delegate permissions to an AWS service](#) in the *IAM User Guide*.
 - **Service-linked role** – 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.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an Amazon EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the Amazon EC2 instance. To assign an AWS role to an Amazon EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the Amazon EC2 instance to get temporary credentials. For more information, see [Using an IAM role to grant permissions to applications running on Amazon EC2 instances](#) in the *IAM User Guide*.

To learn whether to use IAM roles, see [When to create an IAM role \(instead of a user\)](#) 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 is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see [Overview of JSON policies](#) in the *IAM User Guide*.

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**.

Every IAM entity (user or role) starts with no permissions. By default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, role, or group. 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 [Creating IAM policies](#) in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see [Choosing 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 such as an Amazon S3 bucket. Service administrators can use these policies to define what actions a specified principal (account member, user, or role) can perform on that resource and under what conditions. Resource-based policies are inline policies. There are no managed resource-based policies.

Access control lists (ACLs)

Access control lists (ACLs) are a type of policy that controls 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. Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see [Access Control List \(ACL\) overview](#) in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see [Permissions boundaries for IAM entities](#) in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see [How SCPs work](#) in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session

policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. 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 Startups works with IAM

Before you use IAM to manage access to AWS Startups, learn what IAM features are available to use with AWS Startups. To get a high-level view of how AWS Startups and other AWS services work with IAM, see [AWS services that work with IAM](#) in the *IAM User Guide*.

AWS Startups exposes the operation `startups:GetSpendSummary`, which the IDE extension calls to power the AWS credits and cost tracker widget. The IDE extension calls this operation with your own AWS profile credentials. AWS Startups then uses a forward-access session to call AWS Cost Explorer and AWS Billing and Cost Management on your behalf.

Because the operation runs with your identity rather than a separate service role, your identity must have permission for both `startups:GetSpendSummary` and the downstream actions that the operation calls on your behalf: `ce:GetCostAndUsage`, `ce:GetCostForecast`, `billing:GetCredits`, and `aws-portal:ViewBilling`. The `aws-portal:ViewBilling` permission is required in addition to the other three.

No AWS Startups action supports resource-level permissions, so you must set the `Resource` element to `*` (an asterisk). In this case, `Resource: *` is expected. The least-privilege control is the scoped `Action` element. Grant only the specific actions that the operation requires. Do not use wildcard actions such as `startups:*` or `ce:*`.

There is no AWS managed policy for this feature. Use a customer managed or inline identity-based policy that grants only the required actions. For an example, see [AWS Startups identity-based policy examples](#).

Topics

- [AWS Startups Identity-based policies](#)
- [AWS Startups IAM roles](#)

AWS Startups Identity-based policies

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. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you use in a JSON policy, see [IAM JSON policy elements reference](#) in the *IAM User Guide*.

Actions

The `Action` element of an IAM identity-based policy describes the specific action or actions that will be allowed or denied by the policy. Policy actions usually have the same name as the associated AWS API operation. The action is used in a policy to grant permissions to perform the associated operation.

Policy actions in AWS Startups use the `startups:` prefix before the action. For example, to grant permission to call the `GetSpendSummary` operation, include the `startups:GetSpendSummary` action in your policy. Policy statements must include either an `Action` or `NotAction` element. AWS Startups defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows:

```
"Action": [  
    "startups:GetSpendSummary",  
    "ce:GetCostAndUsage",  
    "billing:GetCredits",  
    "aws-portal:ViewBilling"  
]
```

To see a list of AWS Startups actions, see [Actions Defined by AWS Startups](#) in the *IAM User Guide*.

Resources

No AWS Startups action can be performed on a specific resource. For this reason, you must set the `Resource` element to `*` (an asterisk) in your policies.

Condition keys

AWS Startups does not define any service-specific condition keys. You can use the global condition context keys that are available to all AWS services. For the list, see [AWS global condition context keys](#) in the *IAM User Guide*.

Examples

To view examples of AWS Startups identity-based policies, see [AWS Startups identity-based policy examples](#).

AWS Startups IAM roles

An [IAM role](#) is an entity within your AWS account that has specific permissions.

Using temporary credentials with AWS Startups

You can use temporary credentials to sign in with federation, assume an IAM role, or to assume a cross-account role. You obtain temporary security credentials by calling AWS STS API operations such as [AssumeRole](#) or [GetFederationToken](#).

AWS Startups supports using temporary credentials.

Service roles and service-linked roles

AWS Startups does not use service roles or service-linked roles. When AWS Startups calls downstream services, such as AWS Cost Explorer and AWS Billing and Cost Management, it uses your own credentials through a forward-access session rather than a role that the service owns.

AWS Startups identity-based policy examples

By default, IAM users and roles don't have permission to create or modify AWS Startups resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform specific API operations on the specified resources they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see [Creating policies on the JSON tab](#) in the *IAM User Guide*.

Topics

- [Policy best practices](#)
- [Allowing access to the AWS credits and cost tracker](#)

Policy best practices

Identity-based policies determine whether someone can create, access, or delete AWS Startups 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 [IAM Access Analyzer policy validation](#) in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or root users in your 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 [Configuring MFA-protected API access](#) in the *IAM User Guide*.

Allowing access to the AWS credits and cost tracker

The following least-privilege identity-based policy allows an identity to call `startups:GetSpendSummary`, which powers the AWS credits and cost tracker widget, along with the downstream AWS Cost Explorer and AWS Billing and Cost Management actions that the operation performs on your behalf. Because the IDE extension calls the operation with your own AWS profile credentials, attach this policy to your identity.

The `startups:GetSpendSummary` action does not support resource-level permissions, so the `Resource` element is set to `*` (an asterisk). This is expected for this action, and the least-privilege control is the scoped list of actions. Do not use wildcard actions such as `startups:*` or `ce:*`.

Note

If your identity is missing one or more of the downstream permissions, the `startups:GetSpendSummary` operation still returns a successful response. However, one or more sections of the response contain an error with the code `ACCESS_DENIED` and a message from the downstream service. If the credits and cost tracker widget shows missing or empty sections, verify that your identity has all of the actions in the following policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowStartupAdvisorCostTracker",
      "Effect": "Allow",
      "Action": [
        "startups:GetSpendSummary",
        "aws-portal:ViewBilling",
        "billing:GetCredits",
        "ce:GetCostAndUsage",
        "ce:GetCostForecast"
      ],
      "Resource": "*"
    }
  ]
}
```

For more information about the `aws-portal:ViewBilling` permission, see [AWS Billing identity-based policy examples](#) in the *AWS Billing User Guide*. For the complete list of actions, resources, and condition keys, see [Actions, resources, and condition keys for AWS Billing](#) and [Actions, resources, and condition keys for AWS Cost Explorer](#) in the *Service Authorization Reference*.

Troubleshooting AWS Startups identity and access

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

Topics

- [I am not authorized to perform an action in AWS Startups](#)
- [I want to allow people outside of my AWS account to access my AWS Startups resources](#)

I am not authorized to perform an action in AWS Startups

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to view the credits and cost tracker but does not have `startups:GetSpendSummary` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
startups:GetSpendSummary
```

In this case, Mateo asks his administrator to update his policies to allow him to call the `startups:GetSpendSummary` action.

I want to allow people outside of my AWS account to access my AWS Startups 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 Startups supports these features, see [How AWS Startups 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 [How IAM roles differ from resource-based policies](#) in the *IAM User Guide*.

Compliance validation for AWS Startups

To learn whether an AWS service is within the scope of specific compliance programs, see *AWS Services in Scope by Compliance Program* and choose the compliance program that you are interested in.

For a list of AWS services in scope of specific compliance programs, see [AWS Services in Scope by Compliance Program](#). For general information, see [AWS Compliance Programs](#).

You can download third-party audit reports using AWS Artifact. For more information, see [Downloading Reports in AWS Artifact](#).

Your compliance responsibility when using AWS Startups is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- [Security and Compliance Quick Start Guides](#) – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.
- [Architecting for HIPAA Security and Compliance Whitepaper](#) – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.
- [AWS Compliance Resources](#) – This collection of workbooks and guides might apply to your industry and location.
- [Evaluating Resources with Rules](#) in the *AWS Config Developer Guide* – AWS Config; assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.

- [AWS Security Hub](#) – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.

Resilience in AWS Startups

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see [AWS Global Infrastructure](#).

Infrastructure security in AWS Startups

As a managed service, AWS Startups is protected by the AWS global network security procedures that are described in the [Amazon Web Services: Overview of Security Processes](#) whitepaper.

You use AWS published API calls to access AWS Startups through the network. Clients must support Transport Layer Security (TLS) 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the [AWS Security Token Service](#) (AWS STS) to generate temporary security credentials to sign requests.

AWS Startups is available in the US East (N. Virginia) (us-east-1) Region only.

AWS Startups supports interface VPC endpoints powered by AWS PrivateLink. An interface VPC endpoint lets you create a private connection between your VPC and AWS Startups without traversing the public internet. This connection lets AWS Startups communicate with the resources in your VPC using private IP addresses, and traffic between your VPC and the service does not leave the Amazon network. To create an interface VPC endpoint, use the endpoint service name `aws.api.global.startups`. For more information about AWS PrivateLink and interface VPC endpoints, see [What is AWS PrivateLink?](#) in the AWS PrivateLink Guide. For steps to create an interface VPC endpoint for AWS Startups, see [AWS Startups and interface VPC endpoints \(AWS PrivateLink\)](#).

Logging AWS Startups API calls using AWS CloudTrail

AWS Startups is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in AWS Startups. CloudTrail captures all API calls for AWS Startups as events. The calls captured include calls from the AWS Startups console and code calls to the AWS Startups API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AWS Startups. 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 the request that was made to AWS Startups, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

AWS Startups information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS Startups, 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 Startups, 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. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

- [Overview for creating a trail](#)
- [CloudTrail supported services and integrations](#)
- [Configuring Amazon SNS notifications for CloudTrail](#)
- [Receiving CloudTrail log files from multiple Regions](#)
- [Receiving CloudTrail log files from multiple accounts](#)

AWS Startups logs API calls to CloudTrail as management events. For example, calls to the `GetSpendSummary` action generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the [CloudTrail userIdentity element](#).

Understanding AWS Startups log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following examples show CloudTrail log entries that demonstrate the `GetSpendSummary` action. In both entries, the event source is `startups.amazonaws.com`, and `GetSpendSummary` is a read-only management event.

The following example shows a successful `GetSpendSummary` call.

```
{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROEXAMPLEID123456:jane-doe",
    "arn": "arn:aws:sts::111122223333:assumed-role/aws-startup-advisor-user/jane-doe",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROEXAMPLEID123456",
        "arn": "arn:aws:iam::111122223333:role/aws-startup-advisor-user",
        "accountId": "111122223333",
        "userName": "aws-startup-advisor-user"
      }
    }
  }
}
```

```

    },
    "attributes": { "creationDate": "2026-01-01T00:00:00Z", "mfaAuthenticated":
"false" }
  }
},
"eventTime": "2026-01-01T00:00:05Z",
"eventSource": "startups.amazonaws.com",
"eventName": "GetSpendSummary",
"awsRegion": "us-east-1",
"sourceIPAddress": "192.0.2.0",
"userAgent": "aws-sdk-js/3.x",
"requestParameters": null,
"responseElements": null,
"requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
"eventID": "b2c3d4e5-6789-01bc-defa-EXAMPLE22222",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"recipientAccountId": "111122223333",
"eventCategory": "Management",
"tlsDetails": {
  "tlsVersion": "TLSv1.3",
  "cipherSuite": "TLS_AES_128_GCM_SHA256",
  "clientProvidedHostHeader": "startups.global.api.aws"
}
}

```

The following example shows a failed `GetSpendSummary` call. This entry shows an action-level authorization failure for `startups:GetSpendSummary`, where no identity-based policy allows the caller to perform the action. This action-level failure differs from the partial-response behavior. In that case, a successful call returns `ACCESS_DENIED` in individual response sections when the caller lacks the downstream Cost Explorer or Billing permissions.

```

{
  "eventVersion": "1.11",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROEXAMPLEID123456:jane-doe",
    "arn": "arn:aws:sts::111122223333:assumed-role/aws-startup-advisor-user/jane-
doe",
    "accountId": "111122223333",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {

```

```

        "sessionIssuer": {
            "type": "Role",
            "principalId": "AROEXAMPLEID123456",
            "arn": "arn:aws:iam::111122223333:role/aws-startup-advisor-user",
            "accountId": "111122223333",
            "userName": "aws-startup-advisor-user"
        },
        "attributes": { "creationDate": "2026-01-01T00:00:00Z", "mfaAuthenticated":
"false" }
    },
    "eventTime": "2026-01-01T00:00:05Z",
    "eventSource": "startups.amazonaws.com",
    "eventName": "GetSpendSummary",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-sdk-js/3.x",
    "errorCode": "AccessDenied",
    "errorMessage": "User: arn:aws:sts::111122223333:assumed-role/aws-startup-advisor-
user/jane-doe is not authorized to perform: startups:GetSpendSummary because no
identity-based policy allows the startups:GetSpendSummary action",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "c3d4e5f6-7890-12cd-efab-EXAMPLE33333",
    "eventID": "d4e5f6a7-8901-23de-fabc-EXAMPLE44444",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "startups.global.api.aws"
    }
}

```

AWS Startups and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and AWS Startups, the service that powers the AWS Startup Advisor credits and cost tracker, by creating an *interface VPC endpoint*. Interface endpoints are powered by [AWS PrivateLink](#), a technology that enables you to privately access the AWS Startups APIs without an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with the AWS Startups APIs. Traffic between your VPC and AWS Startups does not leave the Amazon network.

Each interface endpoint is represented by one or more [Elastic Network Interfaces](#) in your subnets.

For more information, see [Interface VPC endpoints \(AWS PrivateLink\)](#) in the *Amazon VPC User Guide*.

Considerations for AWS Startups VPC endpoints

Before you set up an interface VPC endpoint for AWS Startups, ensure that you review [Interface endpoint properties and limitations](#) in the *Amazon VPC User Guide*.

AWS Startups supports making calls to all of its API actions from your VPC.

The interface VPC endpoint for AWS Startups is available in the US East (N. Virginia) (us-east-1) Region only.

Creating an interface VPC endpoint for AWS Startups

You can create a VPC endpoint for the AWS Startups service using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Creating an interface endpoint](#) in the *Amazon VPC User Guide*.

Create a VPC endpoint for AWS Startups using the following service name:

- `aws.api.global.startups`

If you enable private DNS for the endpoint, you can make API requests to AWS Startups using its DNS name `startups.global.api.aws`.

For more information, see [Accessing a service through an interface endpoint](#) in the *Amazon VPC User Guide*.

Creating a VPC endpoint policy for AWS Startups

You can attach an endpoint policy to your VPC endpoint that controls access to AWS Startups. The policy specifies the following information:

- The principal that can perform actions.
- The actions that can be performed.
- The resources on which actions can be performed.

For more information, see [Controlling access to services with VPC endpoints](#) in the *Amazon VPC User Guide*.

Example: VPC endpoint policy for AWS Startups actions

The following is an example of an endpoint policy for AWS Startups. When attached to an endpoint, this policy grants access to the listed AWS Startups actions for all principals on all resources.

```
{
  "Statement": [
    {
      "Principal": "*",
      "Effect": "Allow",
      "Action": [
        "startups:GetSpendSummary"
      ],
      "Resource": "*"
    }
  ]
}
```