What is the Developer Tools console?

The Developer Tools console is home to a set of services and features that you can use individually or collectively to help you develop software, either individually or as a team. The developer tools can help you securely store, build, test, and deploy your software. Used individually or collectively, these tools provide support for DevOps, continuous integration, and continuous delivery (CI/CD).

The Developer Tools console includes the following services:

- **AWS CodeCommit** is a fully managed source control service that hosts private Git repositories. You can use repositories to privately store and manage assets (such as documents, source code, and binary files) in the AWS Cloud. Your repositories store your project history from the first commit through the latest changes. You can work collaboratively on code in repositories by commenting on code and creating pull requests to help ensure code quality.

- **AWS CodeBuild** is a fully managed build service that compiles your source code, runs unit tests, and produces artifacts that are ready to deploy. It provides prepackaged build environments for popular programming languages and build tools such as Apache Maven, Gradle, and more. You can also customize build environments in CodeBuild to use your own build tools.

- **AWS CodeDeploy** is a fully managed deployment service that automates software deployments to compute services such as Amazon EC2, AWS Lambda, and your on-premises servers. It can help you rapidly release new features, avoid downtime during application deployment, and handle the complexity of updating your applications.

- **AWS CodePipeline** is a continuous integration and continuous delivery service you can use to model, visualize, and automate the steps required to release your software. You can quickly model and configure the different stages of a software release process. You can build, test, and deploy your code every time there is a code change, based on the release process models you define.

Here's an example of how you can use the services in the Developer Tools console together to help you develop software.
In this example, developers create a repository in CodeCommit and use it to develop and collaborate on their code. They create a build project in CodeBuild to build and test their code, and use CodeDeploy to deploy their code to test and production environments. They want to iterate quickly, so they create a pipeline in CodePipeline to detect the changes in the CodeCommit repository. Those changes are built, tests are run, and successfully built and tested code is deployed to the test server. The team adds test stages to the pipeline to run more tests on the staging server, such as integration or load tests. Upon the successful completion of those tests, a team member reviews the results and if satisfied, manually approves the changes for production. CodePipeline deploys the tested and approved code to production instances.

This is just one simple example of how you can use one or more of the services available in the Developer Tools console to help you develop software. Each of the services can be customized to meet your needs. They offer many integrations with other products and services, both in AWS and with other third-party tools. For more information, see the following topics:

- **CodeCommit**: Product and service integrations
- **CodeBuild**: Use CodeBuild with Jenkins
- **CodeDeploy**: Product and service integrations
- **CodePipeline**: Product and service integrations
Are you a first-time user?

If you are a first-time user of one or more of the services available in the Developer Tools console, we recommend that you begin by reading the following topics:

- Getting started with CodeCommit
- Getting started with CodeBuild, Concepts
- Getting started with CodeDeploy, Primary components
- Getting started with CodePipeline, Concepts

Features of the developer tools console

The Developer Tools console includes the following features:

- The Developer Tools console includes a notifications manager feature that you can use to subscribe to events in AWS CodeBuild, AWS CodeCommit, AWS CodeDeploy, and AWS CodePipeline. This feature has its own API, AWS CodeStar Notifications. You can use the notifications feature to quickly notify users about events in the repositories, build projects, deployment applications, and pipelines that are most important to their work. A notifications manager helps make users aware of events that occur on repositories, builds, deployments, or pipelines so that they can quickly take action, such as approving changes or correcting errors. For more information, see What are notifications? (p. 3)
- The Developer Tools console includes a connections feature that you can use to associate your AWS resources with third-party source code providers. This feature has its own API, AWS CodeStar Connections. You can use the connections feature to set up an authorized connection with a third-party provider and use the connection resource with other AWS services. For more information, see What are connections? (p. 40)

What are notifications?

The notifications feature in the Developer Tools console is a notifications manager for subscribing to events in AWS CodeBuild, AWS CodeCommit, AWS CodeDeploy and AWS CodePipeline. It has its own API, AWS CodeStar Notifications. You can use the notifications feature to quickly notify users about events in the repositories, build projects, deployment applications, and pipelines that are most important to their work. A notifications manager helps make users aware of events that occur on repositories, builds, deployments, or pipelines so that they can quickly take action, such as approving changes or correcting errors.

What can I do with notifications?

You can use the notifications feature to create and manage notification rules to notify users of important changes to their resources, including:

- Build successes and failures in CodeBuild build projects.
- Deployment successes and failures in CodeDeploy applications.
- Creation of and updates in pull requests, including comments on code, in CodeCommit repositories.
- Manual approval statuses and pipeline runs in CodePipeline.

You can set up notifications so that they go to user email addresses that are subscribed to an Amazon SNS topic. You can also integrate this feature with AWS Chatbot and have notifications delivered to Slack channels or Amazon Chime chatrooms.
How do notifications work?

When you configure a notification rule for a supported resource, such as a repository, build project, application, or pipeline, the notifications feature creates an Amazon EventBridge rule that monitors for the events you specify. When an event of that type occurs, the notification rule sends notifications to the Amazon SNS topics specified as targets for that rule. Subscribers to those targets receive notifications about those events.

How do I get started with notifications?

To get started, here are some useful topics to review:

- Learn about the concepts (p. 4) for notifications.
- Set up the resources you need (p. 9) to start working with notifications.
- Get started with your first notification rules (p. 13) and receive your first notifications.

Notification concepts

Setting up and using notifications is easier if you understand the concepts and terms. Here are some concepts to know about as you use notifications.

Topics

- Notifications (p. 4)
- Notification rules (p. 5)
- Events (p. 5)
- Detail types (p. 5)
- Targets (p. 6)
- Notifications and AWS CodeStar Notifications (p. 7)
- Events for notification rules on repositories (p. 7)
- Events for notification rules on build projects (p. 7)
- Events for notification rules on deployment applications (p. 8)
- Events for notification rules on pipelines (p. 8)

Notifications

A notification is a message that contains information about events that occur in the resources you and your developers use. You can set up notifications so that users of a resource, such as a build project, repository, deployment application, or pipeline, receive emails about the event types you specify according to the notification rule you create.

Notifications for AWS CodeCommit can contain user identity information, such as a display name or an email address, through the use of session tags. CodeCommit supports the use of session tags, which are key-value pair attributes that you pass when you assume an IAM role, use temporary credentials, or federate a user in AWS Security Token Service (AWS STS). You can also associate tags with an IAM user. CodeCommit includes the values for `displayName` and `emailAddress` in notification content if those tags are present. For more information, see Using tags to provide additional identity information in CodeCommit.

Important

Notifications include project-specific information such as build status, deployment status, lines of code that have comments, and pipeline approvals. Notification content might change as new features are added. As a security best practice, you should regularly review the targets...
of notification rules and the Amazon SNS topic subscribers. For more information, see
Understanding notification contents and security (p. 96).

Notification rules

A notification rule is an AWS resource that you create to specify when and where notifications are sent. It defines:

- The conditions under which a notification is created. These conditions are based on events that you choose, which are specific to the resource type. Supported resource types include build projects in AWS CodeBuild, deployment applications in AWS CodeDeploy, pipelines in AWS CodePipeline, and repositories in AWS CodeCommit.
- The targets to which the notification is sent. You can specify up to 10 targets for a notification rule.

Notification rules are scoped to individual build projects, deployment applications, pipelines, and repositories. Notification rules have both user-defined friendly names and Amazon Resource Names (ARNs). Notification rules must be created in the same AWS Region where the resource exists. For example, if your build project is in the US East (Ohio) Region, your notification rule must be created in the US East (Ohio) Region, too.

You can define up to 10 notification rules for a resource.

Events

An event is a change of state on a resource that you want to monitor. Each resource has a list of event types you can choose from. When you set up a notification rule on a resource, you specify the events that cause notifications to be sent. For example, if you set up notifications for a repository in CodeCommit, and you select Created for both Pull request and Branches and tags, a notification is sent every time a user in that repository creates a pull request, branch, or Git tag.

Detail types

When you create a notification rule, you can choose the level of detail or detail type included in notifications (Full or Basic). The Full setting (the default) includes all information available for the event in the notification, including any enhanced information provided by services for specific events. The Basic setting includes only a subset of the available information.

The following table lists the enhanced information available for specific event types and describes the differences between the detail types.

<table>
<thead>
<tr>
<th>Service</th>
<th>Event</th>
<th>Full includes</th>
<th>Basic does not include</th>
</tr>
</thead>
<tbody>
<tr>
<td>CodeCommit</td>
<td>Comments on commits</td>
<td>All event details and the content of the comment, including any replies or comment threads. It also includes the line number and the line of code upon which the comment was made.</td>
<td>The content of the comment. line number, line of code, or any comment threads.</td>
</tr>
<tr>
<td></td>
<td>Comments on pull requests</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CodeCommit</td>
<td>Pull request created</td>
<td>All event details and the number of files that were added, modified, or deleted in the pull request</td>
<td>No list of files or details about whether the pull request source branch is included</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>Event</th>
<th>Full includes</th>
<th>Basic does not include</th>
</tr>
</thead>
<tbody>
<tr>
<td>CodeCommit</td>
<td>Pull request created</td>
<td>All event details and the number of files that were added, modified, or deleted in the pull request</td>
<td>No list of files or details about whether the pull request source branch is included</td>
</tr>
</tbody>
</table>

## Service | Event | Full includes | Basic does not include
--- | --- | --- | ---
 |  | request in relation to the destination branch. | has added, modified, or deleted files.
CodePipeline | Manual approval needed | All event details and custom data (if configured). The notification also includes a link to the required approval in the pipeline. | No custom data or link.
CodePipeline | Action execution failed | All event details and the content of the error message for the failure. | No error message content.
  
### Targets

A target is a location for receiving notifications from notification rules. The allowed target types are Amazon SNS topics and AWS Chatbot clients configured for Slack channels. Any user subscribed to the target receives notifications about the events that you specify in the notification rule.

If you want to extend the reach of notifications, you can manually configure integration between notifications and AWS Chatbot so that notifications are sent to Amazon Chime chatrooms. You can then choose the Amazon SNS topic that is configured for that AWS Chatbot client as the target for the notification rule. For more information, see To integrate notifications with AWS Chatbot and Amazon Chime (p. 34).

If you choose to use an AWS Chatbot client as a target, you must first create that client in AWS Chatbot. When you choose an AWS Chatbot client as a target for a notification rule, an Amazon SNS topic is configured for that AWS Chatbot client with all the policies required for notifications to be sent to the Slack channel. You don't have to configure any existing Amazon SNS topics for the AWS Chatbot client.

You can choose to create an Amazon SNS topic as a target as part of creating a notification rule (recommended). You can also choose an existing Amazon SNS topic in the same AWS Region as the notification rule, but you must configure it with the required policy. The Amazon SNS topic that you use for a target must be in your AWS account. It also must be in the same AWS Region as the notification rule and the AWS resource for which the rule was created.

For example, if you create a notification rule for a repository in the US East (Ohio) Region, the Amazon SNS topic must also exist in that Region. If you create an Amazon SNS topic as part of creating a notification rule, the topic is configured with the policy required to allow the publication of events to the topic. This is the best method for working with targets and notification rules. If you choose to use an already-existing topic or create one manually, you must configure it with the required permissions before users receive notifications. For more information, see Configure Amazon SNS topics for notifications (p. 11).

**Note**

If you want to use an existing Amazon SNS topic instead of creating a new one, in Targets, choose its ARN. Make sure the topic has the appropriate access policy, and that the subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different
permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

Notifications and AWS CodeStar Notifications

While a feature of the Developer Tools console, notifications has its own API, AWS CodeStar Notifications. It also has its own AWS resource type (notification rules), permissions, and events. Events for notification rules are logged in AWS CloudTrail. API actions can be allowed or denied through IAM policies.

Events for notification rules on repositories

<table>
<thead>
<tr>
<th>Category</th>
<th>Events</th>
<th>Event IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comments</td>
<td>On commits</td>
<td>codecommit-repository-comments-on-commits</td>
</tr>
<tr>
<td></td>
<td>On pull requests</td>
<td>codecommit-repository-comments-on-pull-requests</td>
</tr>
<tr>
<td>Approvals</td>
<td>Status changed</td>
<td>codecommit-repository-approvals-status-changed</td>
</tr>
<tr>
<td></td>
<td>Rule override</td>
<td>codecommit-repository-approvals-rule-override</td>
</tr>
<tr>
<td>Pull request</td>
<td>Created</td>
<td>codecommit-repository-pull-request-created</td>
</tr>
<tr>
<td></td>
<td>Source updated</td>
<td>codecommit-repository-pull-request-source-updated</td>
</tr>
<tr>
<td></td>
<td>Status changed</td>
<td>codecommit-repository-pull-request-status-changed</td>
</tr>
<tr>
<td></td>
<td>Merged</td>
<td>codecommit-repository-pull-request-merged</td>
</tr>
<tr>
<td>Branches and tags</td>
<td>Created</td>
<td>codecommit-repository-branches-and-tags-created</td>
</tr>
<tr>
<td></td>
<td>Deleted</td>
<td>codecommit-repository-branches-and-tags-deleted</td>
</tr>
<tr>
<td></td>
<td>Updated</td>
<td>codecommit-repository-branches-and-tags-updated</td>
</tr>
</tbody>
</table>

Events for notification rules on build projects

<table>
<thead>
<tr>
<th>Category</th>
<th>Events</th>
<th>Event IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build state</td>
<td>Failed</td>
<td>codebuild-project-build-state-failed</td>
</tr>
<tr>
<td></td>
<td>Succeeded</td>
<td></td>
</tr>
<tr>
<td>Category</td>
<td>Events</td>
<td>Event IDs</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>In-progress</td>
<td></td>
<td>codebuild-project-build-state-succeeded</td>
</tr>
<tr>
<td></td>
<td></td>
<td>codebuild-project-build-state-in-progress</td>
</tr>
<tr>
<td></td>
<td></td>
<td>codebuild-project-build-state-stopped</td>
</tr>
<tr>
<td>Stopped</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stoped</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Build phase</td>
<td>Failure</td>
<td>codebuild-project-build-phase-failure</td>
</tr>
<tr>
<td></td>
<td>Success</td>
<td>codebuild-project-build-phase-success</td>
</tr>
</tbody>
</table>

**Events for notification rules on deployment applications**

<table>
<thead>
<tr>
<th>Category</th>
<th>Events</th>
<th>Event IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deployment</td>
<td>Failed</td>
<td>codedeploy-application-deployment-failed</td>
</tr>
<tr>
<td></td>
<td>Succeeded</td>
<td>codedeploy-application-deployment-succeeded</td>
</tr>
<tr>
<td></td>
<td>Started</td>
<td>codedeploy-application-deployment-started</td>
</tr>
</tbody>
</table>

**Events for notification rules on pipelines**

<table>
<thead>
<tr>
<th>Category</th>
<th>Events</th>
<th>Event IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action execution</td>
<td>Succeeded</td>
<td>codepipeline-pipeline-action-execution-succeeded</td>
</tr>
<tr>
<td></td>
<td>Failed</td>
<td>codepipeline-pipeline-action-execution-failed</td>
</tr>
<tr>
<td></td>
<td>Canceled</td>
<td>codepipeline-pipeline-action-execution-cancelled</td>
</tr>
<tr>
<td></td>
<td>Started</td>
<td>codepipeline-pipeline-action-execution-started</td>
</tr>
<tr>
<td>Stage execution</td>
<td>Started</td>
<td>codepipeline-pipeline-stage-execution-started</td>
</tr>
<tr>
<td></td>
<td>Succeeded</td>
<td>codepipeline-pipeline-stage-execution-succeeded</td>
</tr>
<tr>
<td></td>
<td>Resumed</td>
<td>codepipeline-pipeline-stage-execution-resumed</td>
</tr>
<tr>
<td></td>
<td>Canceled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Failed</td>
<td></td>
</tr>
</tbody>
</table>
### Setting up

If you have a managed policy for AWS CodeBuild, AWS CodeCommit, AWS CodeDeploy, or AWS CodePipeline applied to your IAM user or role, you have the permissions required to work with notifications within the limitations of the roles and permissions provided by the policy. For example, users who have the `AWSCodeBuildAdminAccess`, `AWSCodeCommitFullAccess`, `AWSCodeDeployFullAccess`, or `AWSCodePipeline_FullAccess` managed policy applied have full administrative access to notifications.

For more information, including example policies, see Identity-based policies (p. 101).

If you have one of these policies applied to your IAM user or role, and a build project in CodeBuild, a repository in CodeCommit, a deployment application in CodeDeploy, or a pipeline in CodePipeline, you are ready to create your first notification rule. Continue to Getting started with notifications (p. 13). If not, see the following topics:

- CodeBuild: Getting started with CodeBuild
- CodeCommit: Getting started with CodeCommit
- CodeDeploy: Tutorials
- CodePipeline: Getting started with CodePipeline

<table>
<thead>
<tr>
<th>Category</th>
<th>Events</th>
<th>Event IDs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipeline execution</td>
<td>Failed</td>
<td>codepipeline-pipeline-pipeline-execution-failed</td>
</tr>
<tr>
<td></td>
<td>Canceled</td>
<td>codepipeline-pipeline-pipeline-execution-canceled</td>
</tr>
<tr>
<td></td>
<td>Started</td>
<td>codepipeline-pipeline-pipeline-execution-started</td>
</tr>
<tr>
<td></td>
<td>Resumed</td>
<td>codepipeline-pipeline-pipeline-execution-resumed</td>
</tr>
<tr>
<td></td>
<td>Succeeded</td>
<td>codepipeline-pipeline-pipeline-execution-succeeded</td>
</tr>
<tr>
<td></td>
<td>Superseded</td>
<td>codepipeline-pipeline-pipeline-execution-superseded</td>
</tr>
<tr>
<td>Manual approval</td>
<td>Failed</td>
<td>codepipeline-pipeline-manual-approval-failed</td>
</tr>
<tr>
<td></td>
<td>Needed</td>
<td>codepipeline-pipeline-manual-approval-needed</td>
</tr>
<tr>
<td></td>
<td>Succeeded</td>
<td>codepipeline-pipeline-manual-approval-succeeded</td>
</tr>
</tbody>
</table>
If you want to manage administrative permissions for notifications for IAM users, groups, or roles yourself, follow the procedures in this topic to set up the permissions and resources you need to use the service.

If you want to use previously created Amazon SNS topics for notifications instead of creating topics specifically for notifications, you must configure an Amazon SNS topic to use as the target for a notification rule by applying a policy that allows events to be published to that topic.

**Note**
To perform the following procedures, you must be signed in with an account that has administrative permissions. For more information, see Creating your first IAM admin user and group.

**Topics**
- Create and apply a policy for administrative access to notifications (p. 10)
- Configure Amazon SNS topics for notifications (p. 11)
- Subscribe users to Amazon SNS topics that are targets (p. 13)

## Create and apply a policy for administrative access to notifications

You can administer notifications by signing in with an IAM user or using a role that has permissions to access the service and the services (AWS CodeBuild, AWS CodeCommit, AWS CodeDeploy, or AWS CodePipeline) for which you want to create notifications. You can also create your own policies and apply them to users or groups.

The following procedure shows you how to configure an IAM group with permissions for administering notifications and adding IAM users. If you do not want to set up a group, you can apply this policy directly to IAM users or to an IAM role that can be assumed by users. You can also use the managed policies for CodeBuild, CodeCommit, CodeDeploy, or CodePipeline, which include policy-appropriate access to notification features depending on the scope of the policy.

### To set up a group with permissions to administer AWS CodeStar Notifications

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
   Make sure you sign in using an account that has administrative permissions.
2. In the IAM console, choose Policies, and then choose Create policy.
3. In Create policy, choose JSON, and paste the following policy statement:

   ```json
   {
   "Version": "2012-10-17",
   "Statement": [
   {
   "Sid": "AWSCodeStarNotificationsFullAccess",
   "Effect": "Allow",
   "Action": ["codestar-notifications:CreateNotificationRule",
   "codestar-notifications:DeleteNotificationRule",
   "codestar-notifications:DescribeNotificationRule",
   "codestar-notifications:ListNotificationRules",
   "codestar-notifications:UpdateNotificationRule",
   "codestar-notifications:Subscribe",
   "codestar-notifications:Unsubscribe",
   "codestar-notifications:DeleteTarget",
   "codestar-notifications:ListTargets",
   "codestar-notifications:ListTagsforResource",
   ```
For examples of other policy permission statements for AWS CodeStar Notifications, see Example: A contributor-level policy for using AWS CodeStar Notifications (p. 119) and Example: A read-only-level policy for using AWS CodeStar Notifications (p. 119).

4. Choose **Review policy**.
5. In **Review policy** section, enter a name (for example, `AWSCodeStarNotificationsFullAccess`) and an optional description for this policy. The description helps you remember the purpose of the policy (for example, *This policy provides full access to AWS CodeStar Notifications.*)
6. Choose **Create policy**.
7. In the navigation bar, choose **Groups**, and then choose **Create group**.
8. In **Group name**, enter a name (for example, `AWSCodeStarNotificationAdmins`) and then choose **Next Step**.
9. In **Attach Policy**, attach the `AWSCodeStarNotificationsFullAccess` policy you just created. Choose **Next Step**.
10. In **Review**, choose **Create Group**.
11. Choose the group name from the list. In **Users**, choose **Add Users to Group**. Add the IAM users you want to have full administrative access for AWS CodeStar Notifications.

### Configure Amazon SNS topics for notifications

The easiest way to set up notifications is to create an Amazon SNS topic when you create a notification rule. You can use an existing Amazon SNS topic if it meets the following requirements:

- It was created in the same AWS Region as the resource (build project, deployment application, repository, or pipeline) for which you want to create notification rules.
- It has not been used for sending notifications for CodeCommit before November 5, 2019. If it has, it will contain policy statements that enabled that functionality. You can choose to use this topic, but you will need to add the additional policy as specified in the procedure. You should not remove the existing policy statement if one or more repositories is still configured for notifications before November 5, 2019.
- It has a policy that allows AWS CodeStar Notifications to publish notifications to the topic.

**To configure an Amazon SNS topic to use as a target for AWS CodeStar Notifications notification rules**

2. In the navigation bar, choose **Topics**, choose the topic you want to configure, and then choose **Edit**.
3. Expand **Access policy**, and then choose **Advanced**.
4. In the JSON editor, add the following statement to the policy. Include the topic ARN, AWS Region, AWS account ID, and topic name.

```json
{
    "Sid": "AWSCodeStarNotifications_publish",
    "Effect": "Allow",
    "Resource": "*"
}]
```
"Principal": {
  "Service": [
    "codestar-notifications.amazonaws.com"
  ]
},
"Action": "SNS:Publish",
"Condition": {
  "StringEquals": {
    "aws:SourceAccount": "123456789012"
  }
}
}

The policy statement should look like the following.

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "SNS:GetTopicAttributes",
        "SNS:SetTopicAttributes",
        "SNS:AddPermission",
        "SNS:RemovePermission",
        "SNS:DeleteTopic",
        "SNS:Subscribe",
        "SNS:ListSubscriptionsByTopic",
        "SNS:Publish",
        "SNS:Receive"
      ],
      "Condition": {
        "StringEquals": {
          "AWS:SourceOwner": "123456789012"
        }
      }
    },
    {
      "Sid": "AWSCodeStarNotifications_publish",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "codestar-notifications.amazonaws.com"
        ]
      },
      "Action": "SNS:Publish",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```
5. Choose **Save changes**.

6. If you want to use an AWS KMS-encrypted Amazon SNS topic to send notifications, you must also enable compatibility between the event source (AWS CodeStar Notifications) and the encrypted topic by adding the following statement to the policy of the AWS KMS key. Replace the AWS Region (in this example, us-east-2) with the AWS Region where the key was created.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": {
            "Service": "codestar-notifications.amazonaws.com"
         },
         "Action": [
            "kms:GenerateDataKey*",
            "kms:Decrypt"
         ],
         "Resource": "*",
         "Condition": {
            "StringEquals": {
               "kms:ViaService": "sns.us-east-2.amazonaws.com"
            }
         }
      }
   ]
}
```

For more information, see [Encryption at rest](https://aws.amazon.com/documentation/kms/encryption-at-rest/) and [Using policy conditions with AWS KMS](https://aws.amazon.com/documentation/kms/) in the *AWS Key Management Service Developer Guide*.

### Subscribe users to Amazon SNS topics that are targets

Before users can receive notifications, they must be subscribed to the Amazon SNS topic that is the target of the notification rule. If users are subscribed by email address, they must confirm their subscription before they receive notifications. To send notifications to users in Slack chatrooms or Amazon Chime chatrooms, see [Configure integration between notifications and AWS Chatbot](p. 33).

#### To subscribe users to an Amazon SNS topic used for notifications

2. In the navigation bar, choose **Topics**, and then choose the topic to which you want to subscribe users.
3. In **Subscriptions**, choose **Create subscription**.
4. In **Protocol**, choose **Email**. In **Endpoint**, enter the email address, and then choose **Create subscription**.

### Getting started with notifications

The easiest way to get started with notifications is to set up a notification rule on one of your build projects, deployment applications, pipelines, or repositories.
**Note**
The first time you create a notification rule, a service-linked role is created in your account. For more information, see Using service-linked roles for AWS CodeStar Notifications (p. 129).

**Topics**
- Prerequisites (p. 14)
- Create a notification rule for a repository (p. 14)
- Create a notification rule for a build project (p. 15)
- Create a notification rule for a deployment application (p. 16)
- Create a notification rule for a pipeline (p. 17)

**Prerequisites**
Complete the steps in Setting up (p. 9). You also need a resource for which you create a notification rule.

- Create a build project in CodeBuild or use an existing one.
- Create an application or use an existing deployment application.
- Create a pipeline in CodePipeline or use an existing one.
- Create an AWS CodeCommit repository or use an existing one.

**Create a notification rule for a repository**
You can create notification rules to send notifications about repository events that are important to you. The following steps show you how to set up a notification rule on a single repository event. These steps are written with the assumption that you have a repository configured in your AWS account.

**Important**
If you set up notifications in CodeCommit before November 5, 2019, the Amazon SNS topics used for those notifications will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

2. Choose a repository from the list and open it.
3. Choose Notify, and then choose Create notification rule. You can also choose Settings, choose Notifications, and then choose Create notification rule.
4. In Notification name, enter a name for the rule.
5. In Detail type, choose Basic if you want only the information provided to Amazon EventBridge included in the notification. Choose Full if you want to include information provided to Amazon EventBridge and information that might be supplied by the resource service or the notification manager.

   For more information, see Understanding notification contents and security (p. 96).
6. In Events that trigger notifications, under Branches and tags, select Created.
7. In Targets, choose Create SNS topic.

**Note**
When you create the topic as part of creating the notification rule, the policy that allows CodeCommit to publish events to the topic is applied for you. Using a topic created for
notification rules helps ensure that you subscribe only those users who you want to receive notifications about this repository.

After the `codestar-notifications-` prefix, enter a name for the topic, and then choose **Submit**.

**Note**
If you want to use an existing Amazon SNS topic instead of creating a new one, in **Targets**, choose its ARN. Make sure the topic has the appropriate access policy, and that the subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see [Configure Amazon SNS topics for notifications](p. 11) and [Understanding notification contents and security](p. 96).

8. Choose **Submit**, and then review the notification rule.
9. Subscribe your email address to the Amazon SNS topic you just created. For more information, see [To subscribe users to an Amazon SNS topic used for notifications](p. 13).
10. Navigate to your repository and create a test branch from the default branch.
11. After you create the branch, the notification rule sends a notification to all topic subscribers with information about that event.

### Create a notification rule for a build project

You can create notification rules to send notifications about the events on your build project that are important to you. The following steps show you how to set up a notification rule on a single build project event. These steps are written with the assumption that you have a build project configured in your AWS account.

2. Choose a build project from the list and open it.
3. Choose **Notify**, and then choose **Create notification rule**. You can also choose **Settings**, and then choose **Create notification rule**.
4. In **Notification name**, enter a name for the rule.
5. In **Detail type**, choose **Basic** if you want only the information provided to Amazon EventBridge included in the notification. Choose **Full** if you want to include information provided to Amazon EventBridge and information that might be supplied by the resource service or the notification manager.

For more information, see [Understanding notification contents and security](p. 96).

6. In **Events that trigger notifications**, under **Build phase**, select **Success**.
7. In **Targets**, choose **Create SNS topic**.

**Note**
When you create the topic as part of creating the notification rule, the policy that allows CodeBuild to publish events to the topic is applied for you. Using a topic created for notification rules helps ensure that you subscribe only those users you want to receive notifications about this build project.

After the `codestar-notifications-` prefix, enter a name for the topic, and then choose **Submit**.

**Note**
If you want to use an existing Amazon SNS topic instead of creating a new one, in **Targets**, choose its ARN. Make sure the topic has the appropriate access policy, and that the
Getting started with notifications

The subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

8. Choose Submit, and then review the notification rule.
9. Subscribe your email address to the Amazon SNS topic you just created. For more information, see To subscribe users to an Amazon SNS topic used for notifications (p. 13).
10. Navigate to your build project and start a build.
11. After the build phase is successfully completed, the notification rule sends a notification to all topic subscribers with information about that event.

Create a notification rule for a deployment application

You can create notification rules to send notifications about the events on your deployment application that are important to you. The following steps show you how to set up a notification rule on a single build project event. These steps are written with the assumption that you have a deployment application configured in your AWS account.

2. Choose an application from the list and open it.
3. Choose Notify, and then choose Create notification rule. You can also choose Settings, and then choose Create notification rule.
4. In Notification name, enter a name for the rule.
5. In Detail type, choose Basic if you want only the information provided to Amazon EventBridge included in the notification. Choose Full if you want to include information provided to Amazon EventBridge and information that might be supplied by the resource service or the notification manager.

For more information, see Understanding notification contents and security (p. 96).

6. In Events that trigger notifications, under Deployment, select Succeeded.
7. In Targets, choose Create SNS topic.

**Note**
When you create the topic as part of creating the notification rule, the policy that allows CodeDeploy to publish events to the topic is applied for you. Using a topic created for notification rules helps ensure that you subscribe only those users you want to receive notifications about this deployment application.

After the codestar-notifications- prefix, enter a name for the topic, and then choose Submit.

**Note**
If you want to use an existing Amazon SNS topic instead of creating a new one, in Targets, choose its ARN. Make sure the topic has the appropriate access policy, and that the subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the
one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

8. Choose Submit, and then review the notification rule.

9. Subscribe your email address to the Amazon SNS topic you just created. For more information, see To subscribe users to an Amazon SNS topic used for notifications (p. 13).

10. Navigate to your deployment application and start a deployment.

11. After the deployment succeeds, the notification rule sends a notification to all topic subscribers with information about the event.

Create a notification rule for a pipeline

You can create notification rules to send notifications about the events on your pipeline that are important to you. The following steps show you how to set up a notification rule on a single pipeline event. These steps are written with the assumption that you have a pipeline configured in your AWS account.

2. Choose a pipeline from the list and open it.
3. Choose Notify, and then choose Create notification rule. You can also choose Settings, and then choose Create notification rule.
4. In Notification name, enter a name for the rule.
5. In Detail type, choose Basic if you want only the information provided to Amazon EventBridge included in the notification. Choose Full if you want to include information provided to Amazon EventBridge and information that might be supplied by the resource service or the notification manager.
6. For more information, see Understanding notification contents and security (p. 96).
7. In Events that trigger notifications, under Action execution, select Started.
8. In Targets, choose Create SNS topic.
9. Note
   When you create the topic as part of creating the notification rule, the policy that allows CodePipeline to publish events to the topic is applied for you. Using a topic created for notification rules helps ensure that you subscribe only those users you want to receive notifications about this pipeline.
   After the codestar-notifications- prefix, enter a name for the topic, and then choose Submit.
   Note
   If you want to use an existing Amazon SNS topic instead of creating a new one, in Targets, choose its ARN. Make sure the topic has the appropriate access policy, and that the subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).
10. Choose Submit, and then review the notification rule.
11. Subscribe your email address to the Amazon SNS topic you just created. For more information, see To subscribe users to an Amazon SNS topic used for notifications (p. 13).
12. Navigate to your pipeline, and then choose Release change.
11. When the action starts, the notification rule sends a notification to all topic subscribers with information about the event.

Working with notification rules

A notification rule is where you configure which events you want users to receive notifications about and specify the targets that receive those notifications. You can send notifications directly to users through Amazon SNS, or through AWS Chatbot clients configured for Slack channels. If you want to extend the reach of notifications, you can manually configure integration between notifications and AWS Chatbot so that notifications are sent to Amazon Chime chatrooms. For more information, see Targets (p. 6) and To integrate notifications with AWS Chatbot and Amazon Chime (p. 34).
Create notification rule

Notification rules set up a subscription to events that happen with your resources. When these events occur, they trigger notifications sent to the targets you designate. You can manage your notification preferences in Settings.

Notification rule settings

**Notification name**
MyNotificationRuleForPullRequests

**Detail type**
Choose the level of detail you want in notifications. Learn more about notifications and security.

- **Full**
  - Includes any supplemental information about events provided by the resource or the notifications feature.

- **Basic**
  - Includes only information provided in resources.

Events that trigger notifications

**Comments**
- [ ] On commits
- [x] On pull requests

**Approvals**
- [ ] Status changed
- [x] Rule override

**Pull request**
- [x] Source updated
- [ ] Created
- [x] Created
- [ ] Status changed
- [x] Merged

**Branches and tags**
- [ ] Created
- [ ] Deleted
- [ ] Updated

Targets

Choose a target type for the notification rule. SNS topics can be created specifically for use with the notification system. Existing topics can be modified for use with notifications. AWS Chatbot clients for Slack integration must be configured before you can choose them as a target type. Learn more.
You can use the Developer Tools console or the AWS CLI to create and manage notification rules.

**Topics**
- Create a notification rule (p. 20)
- View notification rules (p. 23)
- Edit a notification rule (p. 24)
- Enable or disable notifications for a notification rule (p. 26)
- Delete a notification rule (p. 26)

**Create a notification rule**

You can use the Developer Tools console or the AWS CLI to create notification rules. You can create an Amazon SNS topic to use as a target for a notification rule as part of creating the rule. If you want to use an AWS Chatbot client as a target, you must create that client before you can create the rule. For more information, see Configure an AWS Chatbot client for a slack channel (p. 33).

**To create a notification rule (console)**

2. Use the navigation bar to navigate to the resource.
   - For CodeBuild, choose **Build**, choose **Build projects**, and choose a build project.
   - For CodeCommit, choose **Source**, choose **Repositories**, and choose a repository.
   - For CodeDeploy, choose **Applications**, and choose an application.
   - For CodePipeline, choose **Pipeline**, choose **Pipelines**, and choose a pipeline.
3. On the resource page, choose **Notify**, and then choose **Create notification rule**. You can also go to the **Settings** page for the resource, go to **Notifications** or **Notification rules**, and then choose **Create notification rule**.
4. In **Notification name**, enter a name for the rule.
5. In **Detail type**, choose **Basic** if you want only the information provided to Amazon EventBridge included in the notification. Choose **Full** if you want to include information provided to Amazon EventBridge and information that might be supplied by the resource service or the notification manager.
   - For more information, see Understanding notification contents and security (p. 96).
6. In **Events that trigger notifications**, select the events for which you want to send notifications. For event types for a resource, see the following:
   - CodeBuild: Events for notification rules on build projects (p. 7)
   - CodeCommit: Events for notification rules on repositories (p. 7)
   - CodeDeploy: Events for notification rules on deployment applications (p. 8)
   - CodePipeline: Events for notification rules on pipelines (p. 8)
7. In **Targets**, do one of the following:
   - If you have already configured a resource to use with notifications, in **Choose target type**, choose either **AWS Chatbot (Slack)** or **SNS topic**. In **Choose target**, choose the name of the client (for a Slack client configured in AWS Chatbot) or the Amazon Resource Name (ARN) of the Amazon SNS topic (for Amazon SNS topics already configured with the policy required for notifications).
   - If you have not configured a resource to use with notifications, choose **Create target**, and then choose **SNS topic**. Provide a name for the topic after **codestar-notifications-**, and then choose **Create**.
Note

- If you create the Amazon SNS topic as part of creating the notification rule, the policy that allows the notifications feature to publish events to the topic is applied for you. Using a topic created for notification rules helps ensure that you subscribe only those users that you want to receive notifications about this resource.

- You cannot create an AWS Chatbot client as part of creating a notification rule. If you choose AWS Chatbot (Slack), you will see a button directing you to configure a client in AWS Chatbot. Choosing that option opens the AWS Chatbot console. For more information, see Configure an AWS Chatbot client for a slack channel (p. 33).

- If you want to use an existing Amazon SNS topic as a target, you must add the required policy for AWS CodeStar Notifications in addition to any other policies that might exist for that topic. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

8. Choose Submit, and then review the notification rule.

Note
Users must subscribe and confirm subscriptions to the Amazon SNS topic you specified as the target of the rule before they will receive notifications. For more information, see To subscribe users to an Amazon SNS topic used for notifications (p. 13).

To create a notification rule (AWS CLI)

1. At a terminal or command prompt, run the create-notification rule command to generate the JSON skeleton.

   ```bash
   aws codestar-notifications create-notification-rule --generate-cli-skeleton > rule.json
   ```

   You can name the file anything you want. In this example, the file is named rule.json.

2. Open the JSON file in a plaintext editor and edit it to include the resource, event types, and Amazon SNS target that you want for the rule.

   The following example shows a notification rule named MyNotificationRule for a repository named MyDemoRepo in an AWS account with the ID 123456789012. Notifications with the full detail type are sent to an Amazon SNS topic named MyNotificationTopic when branches and tags are created.

   ```json
   {
       "Name": "MyNotificationRule",
       "EventTypeIds": [
           "codecommit-repository-branches-and-tags-created"
       ],
       "Targets": [
           {
               "TargetType": "SNS",
               "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic"
           }
       ],
       "Status": "ENABLED",
       "DetailType": "FULL"
   }
   ```
Save the file.

3. Using the file you just edited, at the terminal or command line, run the `create-notification-rule` command again to create the notification rule.

```bash
aws codestar-notifications create-notification-rule --cli-input-json file://rule.json
```

4. If successful, the command returns the ARN of the notification rule, similar to the following.

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE"
}
```

**To list event types for notification rules (AWS CLI)**

1. At a terminal or command prompt, run the `list-event-types` command. You can use the `--filters` option to limit the response to a specific resource type or other attribute. For example, the following returns a list of event types for CodeDeploy applications.

```bash
aws codestar-notifications list-event-types --filters Name=SERVICE_NAME,Value=CodeDeploy
```

2. This command produces output similar to the following.

```
{
    "EventTypes": [
        {
            "EventTypeId": "codedeploy-application-deployment-succeeded",
            "ServiceName": "CodeDeploy",
            "EventTypeName": "Deployment: Succeeded",
            "ResourceType": "Application"
        },
        {
            "EventTypeId": "codedeploy-application-deployment-failed",
            "ServiceName": "CodeDeploy",
            "EventTypeName": "Deployment: Failed",
            "ResourceType": "Application"
        },
        {
            "EventTypeId": "codedeploy-application-deployment-started",
            "ServiceName": "CodeDeploy",
            "EventTypeName": "Deployment: Started",
            "ResourceType": "Application"
        }
    ]
}
```

**To add a tag to a notification rule (AWS CLI)**

1. At a terminal or command prompt, run the `tag-resource` command. For example, use the following command to add a tag key-value pair that has the name `Team` and the value `Li_Juan`.

```bash
aws codestar-notifications tag-resource --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/fe1efd35-EXAMPLE --tags Team=Li_Juan
```

2. This command produces output similar to the following.
View notification rules

You can use the Developer Tools console or the AWS CLI to view all of the notification rules for all resources in an AWS Region. You can also view the details of each notification rule. Unlike the process for creating a notification rule, you do not have to go to the resource page for the resource.

To view notification rules (console)
2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rules, review the list of rules configured for your resources in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region.
4. To view the details of a notification rule, choose it from the list, and then choose View details. You can also simply choose its name in the list.

To view a list of notification rules (AWS CLI)
1. At a terminal or command prompt, run the list-notification-rules command to view all notification rules for the specified AWS Region.

   ```bash
   aws codestar-notifications list-notification-rules --region us-east-1
   ```
2. If successful, this command returns the ID and ARN for each notification rule in the AWS Region, similar to the following.

   ```json
   }
   ```

To view details of a notification rule (AWS CLI)
1. At a terminal or command prompt, run the describe-notification-rule command, specifying the ARN of the notification rule.
Working with notification rules

```bash
aws codestar-notifications describe-notification-rule --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE
```

2. If successful, the command returns output similar to the following.

```json
{
    "LastModifiedTimestamp": 1569199844.857,
    "EventTypes": [
        {
            "ServiceName": "CodeCommit",
            "EventTypeName": "Branches and tags: Created",
            "ResourceType": "Repository",
            "EventTypeId": "codecommit-repository-branches-and-tags-created"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL",
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE",
    "Targets": [
        {
            "TargetStatus": "ACTIVE",
            "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic",
            "TargetType": "SNS"
        }
    ],
    "Name": "MyNotificationRule",
    "CreatedTimestamp": 1569199844.857,
    "CreatedBy": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

To view a list of tags for a notification rule (AWS CLI)

1. At a terminal or command prompt, run the `list-tags-for-resource` command to view all tags for a specified notification rule ARN.

```bash
aws codestar-notifications list-tags-for-resource --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/fe1efd35-EXAMPLE
```

2. If successful, this command returns output similar to the following.

```json
{
    "Tags": {
        "Team": "Li_Juan"
    }
}
```

Edit a notification rule

You can edit a notification rule to change its name, the events for which it sends notifications, the detail type, or the target or targets to which it sends notifications. You can use the Developer Tools console or the AWS CLI to edit a notification rule.
To edit a notification rule (console)

2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rules, review the rules configured for resources in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region.
4. Choose the rule from the list, and then choose Edit. Make your changes, and then choose Submit.

To edit a notification rule (AWS CLI)

1. At a terminal or command prompt, run the describe-notification-rule command (p. 23) to view the structure of the notification rule.
2. Run the update-notification rule command to generate the JSON skeleton and then save it to a file.

   ```bash
   aws codestar-notifications update-notification-rule --generate-cli-skeleton
   > update.json
   
   You can name the file anything you want. In this example, the file is update.json.
   
   3. Open the JSON file in a plaintext editor and make changes to the rule.
   
   The following example shows a notification rule named MyNotificationRule for a repository named MyDemoRepo in an AWS account with the ID 123456789012. Notifications are sent to an Amazon SNS topic named MyNotificationTopic when branches and tags are created. The rule name is changed to MyNewNotificationRule.

   ```json
   {
     "Name": "MyNewNotificationRule",
     "EventTypeIds": [
       "codecommit-repository-branches-and-tags-created"
     ],
     "Targets": [
       {
         "TargetType": "SNS",
         "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic"
       }
     ],
     "Status": "ENABLED",
     "DetailType": "FULL"
   }
   
   Save the file.
   
   4. Using the file you just edited, at the terminal or command line, run the update-notification-rule command again to update the notification rule.

   ```bash
   aws codestar-notifications update-notification-rule --cli-input-json file://update.json
   
   5. If successful, the command returns the Amazon Resource Name (ARN) of the notification rule, similar to the following.

   ```json
   {"Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE"}
   ```
To remove a tag from a notification rule (AWS CLI)

1. At a terminal or command prompt, run the `untag-resource` command. For example, the following command removes a tag with the name of `Team`.

   ```bash
   ```

2. If successful, this command returns nothing.

See also

- Add or remove a target for a notification rule (p. 31)
- Enable or disable notifications for a notification rule (p. 26)
- Events (p. 5)

Enable or disable notifications for a notification rule

When you create a notification rule, notifications are enabled by default. You do not have to delete the rule to prevent it from sending notifications. You can simply change its notification status.

To change the notification status for a notification rule (console)

2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rules, review the rules configured for resources in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region.
4. Find the notification rule you want to enable or disable, and choose it to display its details.
5. In Notification status, choose the slider to change the status of the rule:
   - Sending notifications: This is the default.
   - Notifications paused: No notifications are sent to the specified targets.

To change notification status for a notification rule (AWS CLI)

1. Follow the steps in To edit a notification rule (AWS CLI) (p. 25) to obtain the JSON for the notification rule.
2. Edit the Status field to ENABLED (default) or DISABLED (no notifications), and then run the `update-notification-rule` command to change the status.

   ```json
   "Status": "ENABLED"
   ```

Delete a notification rule

There can be only 10 notification rules configured for a resource, so consider deleting rules you no longer need. You can use the Developer Tools console or the AWS CLI to delete a notification rule.

Note

You cannot undo the deletion of a notification rule, but you can recreate it. Deleting a notification rule does not delete the target.
To delete a notification rule (console)

2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rules, review the rules configured for resources in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region.
4. Choose the notification rule, and then choose Delete.
5. Type delete, and then choose Delete.

To delete a notification rule (AWS CLI)

1. At a terminal or command prompt, run the delete-notification-rule command, specifying the ARN of the notification rule.

   ```bash
   aws codestar-notifications delete-notification-rule --arn arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE
   ```

2. If successful, the command returns the ARN of the deleted notification rule, similar to the following.

   ```json
   {
       "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE"
   }
   ```

Working with notification rule targets

A notification rule target is a destination that defines where you want notifications to be sent when a notification rule's event conditions are met. You can choose between Amazon SNS topics and AWS Chatbot clients that are configured for Slack channels. You can create an Amazon SNS topic as a target as part of creating a notification rule (recommended). You can also choose an existing Amazon SNS topic in the same AWS Region as the notification rule, but you must configure it with the required policy. If you choose to use an AWS Chatbot client as a target, you must first create that client in AWS Chatbot.

If you want to extend the reach of notifications, you can manually configure integration between notifications and AWS Chatbot so that notifications are sent to Amazon Chime chatrooms. You can then choose the Amazon SNS topic configured for that AWS Chatbot client as the target for the notification rule. For more information, see To integrate notifications with AWS Chatbot and Amazon Chime (p. 34).

You can use the Developer Tools console or the AWS CLI to manage notification targets. You can use the console or the AWS CLI to create and configure Amazon SNS topics and AWS Chatbot clients as targets (p. 6). You can also configure integration between the Amazon SNS topics that you configure as targets and AWS Chatbot. This makes it possible for you to send notifications to Amazon Chime chatrooms. For more information, see Configure integration between notifications and AWS Chatbot (p. 33).

Topics

- Create or configure a notification rule target (p. 28)
- View notification rule targets (p. 30)
- Add or remove a target for a notification rule (p. 31)
- Delete a notification rule target (p. 32)
Create or configure a notification rule target

Notification rule targets are Amazon SNS topics or AWS Chatbot clients configured for Slack channels.

An AWS Chatbot client must be created before you can select a client as a target. When you choose an AWS Chatbot client as a target for a notification rule, an Amazon SNS topic is configured for that AWS Chatbot client with all the policies required for notifications to be sent to the Slack channel. You don't have to configure any existing Amazon SNS topics for the AWS Chatbot client.

You can create Amazon SNS notification rule targets in the Developer Tools console when you create a notification rule. The policy that allows notifications to be sent to that topic is applied for you. This is the easiest way to create a target for a notification rule. For more information, see Create a notification rule (p. 20).

If you use an existing Amazon SNS topic, you must configure it with an access policy that allows the resource to send notifications to that topic. For an example, see Configure Amazon SNS topics for notifications (p. 11).

**Note**
If you want to use an existing Amazon SNS topic instead of creating a new one, in Targets, choose its ARN. Make sure the topic has the appropriate access policy, and that the subscriber list contains only those users who are allowed to see information about the resource. If the Amazon SNS topic is a topic that was used for CodeCommit notifications before November 5, 2019, it will contain a policy that allows CodeCommit to publish to it that contains different permissions than those required for AWS CodeStar Notifications. Using these topics is not recommended. If you want to use one created for that experience, you must add the required policy for AWS CodeStar Notifications in addition to the one that already exists. For more information, see Configure Amazon SNS topics for notifications (p. 11) and Understanding notification contents and security (p. 96).

If you want to extend the reach of notifications, you can manually configure integration between notifications and AWS Chatbot so that notifications are sent to Amazon Chime chatrooms. For more information, see Targets (p. 6) and To integrate notifications with AWS Chatbot and Amazon Chime (p. 34).

**To configure an existing Amazon SNS topic to use as a notification rule target (console)**

2. In the navigation bar, choose Topics. Choose the topic, and then choose Edit.
3. Expand Access policy, and then choose Advanced.
4. In the JSON editor, add the following statement to the policy. Include the topic ARN, AWS Region, AWS account ID, and topic name.

   ```json
   {
     "Sid": "AWSCodeStarNotifications_publish",
     "Effect": "Allow",
     "Principal": {
       "Service": ["codestar-notifications.amazonaws.com"
     ],
     "Action": "SNS:Publish",
     "Condition": {
       "StringEquals": {
         "aws:SourceAccount": "123456789012"
       }
     }
   }```

28
The policy statement should look like the following.

```json
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "SNS:GetTopicAttributes",
        "SNS:SetTopicAttributes",
        "SNS:AddPermission",
        "SNS:RemovePermission",
        "SNS:DeleteTopic",
        "SNS:Subscribe",
        "SNS:ListSubscriptionsByTopic",
        "SNS:Publish",
        "SNS:Receive"
      ],
      "Condition": {
        "StringEquals": {
          "AWS:SourceOwner": "123456789012"
        }
      }
    },
    {
      "Sid": "AWSCodeStarNotifications_publish",
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "codestar-notifications.amazonaws.com"
        ]
      },
      "Action": "SNS:Publish",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

5. Choose **Save changes**.

6. In **Subscriptions**, review the list of topic subscribers. Add, edit, or delete subscribers as appropriate for this notification rule target. Make sure that the subscriber list contains only those users who are allowed to see information about the resource. For more information, see Understanding notification contents and security (p. 96).
To create an AWS Chatbot client with slack to use as a target

1. Follow the instructions in Setting up AWS Chatbot with slack in the AWS Chatbot Administrator Guide. When you do so, consider the following choices for optimal integration with notifications:

   • When creating an IAM role, consider choosing a role name that makes it easy to identify the purpose of this role (for example, AWSCodeStarNotifications-Chatbot-Slack-Role). This can help you identify the purpose of the role in the future.

   • In SNS topics, you don’t have to choose a topic or an AWS Region. When you choose the AWS Chatbot client as a target (p. 6), an Amazon SNS topic with all the required permissions is created and configured for the AWS Chatbot client as part of the notification rule creation process.

2. Complete the client creation process. This client is then available for you to choose as a target when creating notification rules. For more information, see Create a notification rule (p. 20).

   Note
   Do not remove the Amazon SNS topic from the AWS Chatbot client after it has been configured for you. Doing so will prevent notifications from being sent to Slack.

View notification rule targets

You can use Developer Tools console, not the Amazon SNS console to view all of the notification rule targets for all resources in an AWS Region. You can also view the details of a notification rule target.

To view notification rule targets (console)

2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rule targets, review the list of targets used by notification rules in in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region. If the target status shows as Unreachable, you might need to investigate. For more information, see Troubleshooting (p. 37).

To view a list of notification rule targets (AWS CLI)

1. At a terminal or command rule prompt, run the list-targets command to view a list of all notification rule targets for the specified AWS Region:

   ```bash
   aws codestar-notifications list-targets --region us-east-2
   ```

2. If successful, this command returns the ID and ARN for each notification rule in the AWS Region, similar to the following:

   ```json
   {
   "Targets": [
   { "TargetAddress": "arn:aws:sns:us-east-2:123456789012:MySNSTopicForNotificationRules", "TargetType": "SNS", "TargetStatus": "ACTIVE" },
   { "TargetAddress": "arn:aws:chatbot::123456789012:chat-configuration/slack-channel/MySlackChannelClientForMyDevTeam", "TargetStatus": "ACTIVE", "TargetType": "AWSChatbotSlack" }
   ],
   ```
Add or remove a target for a notification rule

You can edit a notification rule to change the target or targets to which it sends notifications. You can use the Developer Tools console or the AWS CLI to change a notification rule's targets.

To change the targets for a notification rule (console)

2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rules, review the list of rules configured for your resources in your AWS account in the AWS Region where you are currently signed in. Use the selector to change the AWS Region.
4. Choose the rule, and then choose Edit.
5. In Targets, do one of the following:
   - To add another target, choose Add Target, and then choose the Amazon SNS topic or AWS Chatbot (Slack) client that you want to add from the list. You can also choose Create SNS topic to create a topic and add it as a target. A notification rule can have up to 10 targets.
   - To remove a target, next to the target you want to remove, choose Remove target.
6. Choose Submit.

To add a target to a notification rule (AWS CLI)

1. At a terminal or command prompt, run the subscribe command to add a target. For example, the following command adds an Amazon SNS topic as a target for a notification rule.

```
```

2. If successful, the command returns the ARN of the updated notification rule, similar to the following.

```
{
  "Arn": "arn:aws:codestar-notifications:us-east-2:123456789012:notificationrule/dc82df7a-EXAMPLE"
}
```

To remove a target from a notification rule (AWS CLI)

1. At a terminal or command prompt, run the the unsubscribe command to remove a target. For example, the following command removes an Amazon SNS topic as a target for a notification rule.

```
```

```
2. If successful, the command returns the ARN of the updated notification rule and information about
the removed target, similar to the following.

```json
{
  "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/dc82df7a-EXAMPLE",
  "TargetAddress": "arn:aws:sns:us-east-1:123456789012:MyNotificationTopic"
}
```

See also

- Edit a notification rule (p. 24)
- Enable or disable notifications for a notification rule (p. 26)

Delete a notification rule target

You can delete a target if it is no longer needed. A resource can only have 10 notification rule targets
configured for it, so deleting unneeded targets can help create room for other targets you might want to
add to that notification rule.

**Note**

Deleting a notification rule target removes the target from all notification rules configured to
use it as a target, but it does not delete the target itself.

**To delete a notification rule target (console)**

1. Open the AWS Developer Tools console at https://console.aws.amazon.com/codesuite/settings/
   notifications.
2. In the navigation bar, expand Settings, and then choose Notification rules.
3. In Notification rule targets, review the list of targets configured for your resources in your AWS
   account in the AWS Region where you are currently signed in. Use the selector to change the AWS
   Region.
4. Choose the notification rule target, and then choose Delete.
5. Type delete, and then choose Delete.

**To delete a notification rule target (AWS CLI)**

1. At a terminal or command prompt, run the delete-target command, specifying the ARN of the
target. For example, the following command deletes a target that uses an Amazon SNS topic.

   ```
   aws codestar-notifications delete-target --target-address arn:aws:sns:us-east-1:123456789012:MyNotificationTopic
   ```

2. If successful, the command returns nothing. If unsuccessful, the command returns an error. The most
common error is that the topic is the target for one or more notification rules.

   An error occurred (ValidationException) when calling the DeleteTarget operation:
   Unsubscribe target before deleting.

   You can use the --force-unsubscribe-all parameter to remove the target from all notification
   rules configured to use it as a target, and then delete the target.
Configure integration between notifications and AWS Chatbot

AWS Chatbot is an AWS service that makes it possible for DevOps and software development teams to use Amazon Chime chat rooms and Slack channels to monitor and respond to operational events in the AWS Cloud. You can configure integration between notification rule targets and AWS Chatbot so that notifications about events appear in the Amazon Chime room or Slack channel you choose. For more information, see the AWS Chatbot documentation.

Before you configure integration with AWS Chatbot, you must configure a notification rule and a rule target. For more information, see Setting up (p. 9) and Create a notification rule (p. 20). You must also configure a Slack channel or an Amazon Chime chatroom in AWS Chatbot. For more information, see the documentation for these services.

Topics
- Configure an AWS Chatbot client for a slack channel (p. 33)
- Configure clients for slack or Amazon Chime manually (p. 33)

Configure an AWS Chatbot client for a slack channel

You can create notification rules that use an AWS Chatbot client as a target. If you create a client for a Slack channel, you can use this client directly as a target in the workflow for creating a notification rule. This is the easiest way to set up notifications that appear in Slack channels.

To create an AWS Chatbot client with slack to use as a target

1. Follow the instructions in Setting up AWS Chatbot with slack in the AWS Chatbot Administrator Guide. When you do so, consider the following choices for optimal integration with notifications:
   - When creating an IAM role, consider choosing a role name that makes it easy to identify the purpose of this role (for example, AWSCodeStarNotifications-Chatbot-Slack-Role). This can help you identify the purpose of the role in the future.
   - In SNS topics, you don't have to choose a topic or an AWS Region. When you choose the AWS Chatbot client as a target (p. 6), an Amazon SNS topic with all the required permissions is created and configured for the AWS Chatbot client as part of the notification rule creation process.
2. Complete the client creation process. This client is then available for you to choose as a target when creating notification rules. For more information, see Create a notification rule (p. 20).

   Note
   Do not remove the Amazon SNS topic from the AWS Chatbot client after it has been configured for you. Doing so will prevent notifications from being sent to Slack.

Configure clients for slack or Amazon Chime manually

You can choose to create the integration between notifications and Slack or Amazon Chime directly. This is the only method available for configuring notifications to Amazon Chime chatrooms. When you configure this integration manually, you create an AWS Chatbot client that uses an Amazon SNS topic that you have previously configured as the target for a notification rule.
To manually integrate notifications with AWS Chatbot and slack

2. Choose Settings, and then choose Notification rules.
3. In Notification rule targets, find and copy the target.

   **Note**
   You can configure more than one notification rule to use the same Amazon SNS topic as its target. This can help you consolidate messaging, but can have unintended consequences if the subscription list is intended for one notification rule or resource.

4. Open the AWS Chatbot console at https://console.aws.amazon.com/chatbot/.
5. Choose Configure new client, and then choose Slack.
6. Choose Configure.
7. Sign in to your Slack workspace.
8. When you are prompted to confirm the choices, choose Allow.
9. Choose Configure new channel.
10. In Configuration details, in Configuration name, enter a name for your client. This is the name that will appear in the list of available targets for the AWS Chatbot (Slack) target type when you create notification rules.
11. In Configure Slack Channel, in Channel type, choose Public or Private, depending on the type of channel with which you want to integrate.
   - In Public channel, choose the name of the Slack channel from the list.
   - In Private channel ID, enter the channel code or URL.
12. In IAM permissions, in Role, choose Create an IAM role using a template. In Policy templates, choose Notification permissions. In Role name, enter a name for this role (for example, AWSCodeStarNotifications-Chatbot-Slack-Role). In Policy templates, choose Notification permissions.
13. In SNS topics, in SNS Region, choose the AWS Region where you created the notification rule target. In SNS topics, choose the name of the Amazon SNS topic that you configured as the notification rule target.

   **Note**
   This step is not necessary if you will create a notification rule using this client as a target.

   **Note**
   If you configured integration with a private channel, you must invite AWS Chatbot to the channel before you will see notifications in that channel. For more information, see the AWS Chatbot documentation.
15. (Optional) To test the integration, make a change in the resource that matches an event type for a notification rule that is configured to use the Amazon SNS topic as its target. For example, if you have a notification rule configured to send notifications when comments are made on a pull request, comment on a pull request and then watch the Slack channel in the browser to see when the notification appears.

To integrate notifications with AWS Chatbot and Amazon Chime

2. Choose Settings, and then choose Notification rules.
3. In Notification rule targets, find and copy the target.
Note
You can configure more than one notification rule to use the same Amazon SNS topic as its target. This can help you consolidate messaging, but can have unintended consequences if the subscription list is for one notification rule or resource.

4. In Amazon Chime, open the chatroom that you want to configure for integration.
5. Choose the gear icon in the upper-right corner, and then choose Manage webhooks.
6. In the Manage webhooks dialog box, choose New, enter a name for the webhook, and then choose Create.
7. Verify that the webhook appears, and then choose Copy webhook URL.
8. Open the AWS Chatbot console at https://console.aws.amazon.com/chatbot/.
9. Choose Configure new client, and then choose Amazon Chime.
10. In Configuration details, in Configuration name, enter a name for your client.
11. In Webhook URL, paste the URL. In Webhook description, provide an optional description.
12. In IAM permissions, in Role, choose Create an IAM role using a template. In Policy templates, choose Notification permissions. In Role name, enter a name for this role (for example, AWSCodeStarNotifications-Chime-Role).
13. In SNS topics, in SNS Region, choose the AWS Region where you created the notification rule target. In SNS topics, choose the name of the Amazon SNS topic you configured as the notification rule target.
15. (Optional) To test the integration, make a change in the resource that matches an event type for a notification rule that is configured to use the Amazon SNS topic as its target. For example, if you have a notification rule configured to send notifications when comments are made on a pull request, comment on a pull request and then watch the Amazon Chime chatroom to see when the notification appears.

Logging AWS CodeStar Notifications API calls with AWS CloudTrail

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

For more information, see the AWS CloudTrail User Guide.

AWS CodeStar Notifications information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS CodeStar Notifications, 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 CodeStar Notifications, 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** and **Receiving CloudTrail log files from multiple accounts**

All AWS CodeStar Notifications actions are logged by CloudTrail and are documented in the [AWS CodeStar Notifications API Reference](https://docs.aws.amazon.com/codestar-notifications/latest/userguide/using-api.html). For example, calls to the `CreateNotificationRule`, `Subscribe` and `ListEventTypes` actions 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](https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/cloudtrail-define-statement.html).

### Understanding 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 example shows a CloudTrail log entry that demonstrates the creation of a notification rule, including both the `CreateNotificationRule` and `Subscribe` actions.

**Note**

Some of the events in notification log file entries might come from the service-linked role `AWSServiceRoleForCodeStarNotifications`.

```json
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major"
    },
    "eventTime": "2019-10-07T21:34:41Z",
    "eventSource": "events.amazonaws.com",
    "eventName": "CreateNotificationRule",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "codestar-notifications.amazonaws.com",
    "userAgent": "codestar-notifications.amazonaws.com",
    "requestParameters": {
        "description": "This rule is used to route CodeBuild, CodeCommit, CodePipeline, and other Developer Tools notifications to AWS CodeStar Notifications",
        "name": "awscodestarnotifications-rule",
        "..."  
    }
}
```
Troubleshooting

The following information might help you troubleshoot common issues with notifications.

Topics
- I get a permissions error when I try to create a notification rule on a resource (p. 38)
- I cannot view notification rules (p. 38)
- I cannot create notification rules (p. 38)
- I am receiving notifications for a resource I can't access (p. 38)
- I am not receiving Amazon SNS notifications (p. 38)
• I am receiving duplicate notifications about events (p. 39)
• I want to understand why a notification target status shows as unreachable (p. 39)
• I want to increase my quotas for notifications and resources (p. 39)

I get a permissions error when I try to create a notification rule on a resource

Make sure that you have sufficient permissions. For more information, see Identity-based policy examples (p. 115).

I cannot view notification rules

Problem: When you are in the Developer Tools console and choose Notifications under Settings, you see a permissions error.

Possible fixes: You might not have the permissions required to view notifications. While most managed policies for AWS Developer Tools services, such as CodeCommit and CodePipeline, include permissions for notifications, services that do not currently support notifications do not include permissions to view them. Alternatively, you might have a custom policy applied to your IAM user or role that does not allow you to view notifications. For more information, see Identity-based policy examples (p. 115).

I cannot create notification rules

You might not have the permissions required to create a notification rule. For more information, see Identity-based policy examples (p. 115).

I am receiving notifications for a resource I can't access

When you create a notification rule and add a target, the notifications feature does not validate whether the recipient has access to the resource. It is possible for you to receive notifications about a resource that you can't access. If you cannot remove yourself, ask to be removed from the subscription list for the target.

I am not receiving Amazon SNS notifications

To troubleshoot problems with the Amazon SNS topic, check the following:

• Make sure that the Amazon SNS topic was created in the same AWS Region as the notification rule.
• Make sure that your email alias is subscribed to the correct topic and that you have confirmed the subscription. For more information, see Subscribing an endpoint to an Amazon SNS topic.
• Verify that the topic policy has been edited to allow AWS CodeStar Notifications to push notifications to that topic. The topic policy should include a statement similar to the following:

```
{
    "Sid": "AWSCodeStarNotifications_publish",
    "Effect": "Allow",
    "Principal": {
        "Service": [
            "codestar-notifications.amazonaws.com"
        ]
    },
    "Action": "SNS:Publish",
    "Condition": {
        "StringEquals": {
```
For more information, see Configure Amazon SNS topics for notifications (p. 11).

I am receiving duplicate notifications about events

Here are the most common reasons for receiving multiple notifications:

- Multiple notification rules that include the same event type have been configured for a resource, and you are subscribed to the Amazon SNS topics that are the targets for those rules. To solve this issue, either unsubscribe from one of the topics or edit the notification rules to remove duplication.
- One or more notification rule targets are integrated with AWS Chatbot and you are receiving notifications in your email inbox and a Slack channel or Amazon Chime chatroom. To solve this issue, consider unsubscribing your email address from the Amazon SNS topic that is the target for the rule and use the Slack channel or Amazon Chime chatroom to view notifications.

I want to understand why a notification target status shows as unreachable

Targets have two possible statuses: Active and Unreachable. Unreachable indicates that notifications were sent to a target, and the delivery was not successful. Notifications continue to be sent to that target, and if successful, the status resets to Active.

The target for a notification rule might become unavailable for one of the following reasons:

- The resource (Amazon SNS topic or AWS Chatbot client) has been deleted. Choose another target for the notification rule.
- The Amazon SNS topic is encrypted, and either the required policy for encrypted topics is missing, or the AWS KMS key has been deleted. For more information, see Configure Amazon SNS topics for notifications (p. 11).
- The Amazon SNS topic does not have the required policy for notifications. Notifications cannot be sent to an Amazon SNS topic unless it has the policy. For more information, see Configure Amazon SNS topics for notifications (p. 11).
- The supporting service for the target (Amazon SNS or AWS Chatbot) might be experiencing issues.

I want to increase my quotas for notifications and resources

Currently, you cannot change any quotas. See Quotas for notifications (p. 39).

Quotas for notifications

The following table lists the quotas (also referred to as limits) for notifications in the Developer Tools console. For information about limits that can be changed, see AWS service quotas.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of notification rules in an AWS account</td>
<td>1000</td>
</tr>
</tbody>
</table>
What are connections?

You can use the connections feature in the Developer Tools console to connect AWS resources such as AWS CodePipeline to external code repositories. This feature has its own API, the AWS CodeStar Connections API reference. Each connection is a resource that you can give to AWS services to connect to a third-party repository, such as BitBucket. For example, you can add the connection in CodePipeline so that it triggers your pipeline when a code change is made to your third-party code repository. Each connection is named and associated with a unique Amazon Resource Name (ARN) that is used to reference the connection.

What can I do with connections?

You can use connections to integrate third-party provider resources with your AWS resources in developer tools, including:

- Connect to a third-party provider, such as Bitbucket, and use the third-party connection as a source integration with your AWS resources, such as CodePipeline.
- Uniformly manage access to your connection across your resources in CodeBuild build projects, CodeDeploy applications, and pipelines in CodePipeline for your third-party provider.
- Use a connection ARN in your stack templates for CodeBuild build projects, CodeDeploy applications, and pipelines in CodePipeline, without the need to reference stored secrets or parameters.

What Third-party Providers Can I Create Connections For?

Connections can associate your AWS resources with the following third-party repositories:

- Bitbucket Cloud
- GitHub
- GitHub Enterprise Cloud
- GitHub Enterprise Server

For an overview of the connections workflow, see Workflow to create or update connections (p. 42).

The steps to create connections for a cloud provider type, such as GitHub, are different from the steps for an installed provider type, such as GitHub Enterprise Server. For the high-level steps to create a connection by provider type, see Working with connections (p. 50).

What AWS Services Integrate With Connections?

You can use connections to integrate your third-party repository with the following AWS services:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of targets for a notification rule</td>
<td>10</td>
</tr>
<tr>
<td>Maximum number of notification rules for a resource</td>
<td>10</td>
</tr>
</tbody>
</table>
How do connections work?

Before you can create a connection, you must first install, or provide access to, the AWS authentication app on your third-party account. After a connection is installed, it can be updated to use this installation. When you create a connection, you provide access to the AWS resource in your third-party account. This allows the connection to access content, such as source repositories, in the third-party account, on behalf of your AWS resources. You can then share that connection with other AWS services to provide secure OAuth connections between the resources.

If you want to create a connection to an installed provider type, such as GitHub Enterprise Server, you first create a host resource using the AWS Management Console.

Connections are owned by the AWS account that creates them. Connections are identified by an ARN containing a connection ID. The connection ID is a UUID that cannot be changed or remapped. Deleting and re-establishing a connection results in a new connection ID, and therefore a new connection ARN. This means that connection ARNs are never reused.

A newly created connection is in a Pending state. A third-party handshake (OAuth flow) process is required to complete setup of the connection and for it to move from Pending to an Available state. After this is complete, a connection is Available and can be used with AWS services, such as CodePipeline.

A newly created host is in a Pending state. A third-party registration process is required to complete setup of the host and for it to move from Pending to an Available state. After this is complete, a host is Available and can be used for connections to installed provider types.

For an overview of the connections workflow, see Workflow to create or update connections (p. 42). For the high-level steps to create a connection by provider type, see Working with connections (p. 50).

Global resources in AWS CodeStar Connections

Connections are global resources, meaning that the resource is replicated across all AWS Regions.

Although the connection ARN format reflects the Region name where it was created, the resource is not constrained to any Region. The Region where the connection resource was created is the Region where connection resource data updates are controlled. Examples of API operations that control updates to
connection resource data include creating a connection, updating an installation, deleting a connection, or tagging a connection.

Host resources for connections are not globally available resources. You use host resources only in the Region where they were created.

- You only have to create a connection once, and then you can use it in any AWS Region.
- If the Region where the connection was created is having issues, this impacts APIs that control connection resource data, but you can still successfully use the connection in every other Region.
- When you list connection resources in the console or CLI, the list shows all connection resources associated with your account across all Regions.
- When you list host resources in the console or CLI, the list shows host resources associated with your account in the selected Region only.
- When a connection with an associated host resource is listed or viewed with the CLI, the output returns the host ARN regardless of the configured CLI Region.

**Workflow to create or update connections**

When you create a connection, you also create or use an existing installation for the auth handshake with the third-party provider.

Connections can have the following states:

- **Pending** - A pending connection is a connection that must be completed (moved to available) before it can be used.
- **Available** - You can use or pass an available connection to other resources and users in your account.
- **Error** - A connection that has an error state is retried automatically. It cannot be used until it is available.

**Workflow: Creating or updating a connection with the CLI, SDK, or AWS CloudFormation**

You use the CreateConnection API to create a connection using the AWS Command Line Interface (AWS CLI), SDK, or AWS CloudFormation. After it is created, the connection is in a pending state. You complete the process by using the console Set up pending connection option. The console prompts you to create an installation or use an existing installation for the connection. You then use the console to complete the handshake and move the connection to an available state by choosing Complete connection on the console.

**Workflow: Creating or updating a connection with the console**

If you are creating a connection to an installed provider type, such as GitHub Enterprise Server, you first create a host. If you are connecting to a cloud provider type, such as Bitbucket, you skip creating the host and continue to creating a connection.

To create or update a connection using the console, you use the CodePipeline edit action page on the console to choose your third-party provider. The console prompts you to create an installation or use an existing installation for the connection, and then use the console to create the connection. The console completes the handshake and moves the connection from pending to an available state automatically.
How do I get started with connections?

To get started, here are some useful topics to review:

- **Learn** about the concepts (p. 43) for connections.
- **Set up** the resources you need (p. 45) to start working with connections.
- **Get started** with your first connections (p. 46) and connect them to a resource.

**Connections concepts**

Setting up and using the connections feature is easier if you understand the concepts and terms. Here are some concepts to know about as you use connections in the Developer Tools console:
An instance of the AWS app on a third-party account. Installing the AWS CodeStar Connector app allows AWS to access resources within the third-party account. An installation can only be edited on the third-party provider's website.

**connection**

An AWS resource used to connect third-party source repositories to other AWS services.

**third-party repository**

A repository that is provided by a service or company that is not part of AWS. For example, a BitBucket repository is a third-party repository.

**provider type**

A service or company that provides the third-party source repository you want to connect to. You connect your AWS resources to external *provider types*. A provider type where the source repository is installed on the network and infrastructure is an installed provider type. For example, GitHub Enterprise Server is an installed provider type.

**host**

A resource that represents the infrastructure where a third-party provider is installed. Connections use the host to represent the server where your third-party provider is installed, such as GitHub Enterprise Server. You create one host for all connections to that provider type.

**Note**

When you use the console to create a connection to GitHub Enterprise Server, the console creates a host resource for you as part of the process.

## AWS CodeStar Connections supported providers and versions

This chapter provides information about the providers and versions that AWS CodeStar Connections supports.

**Topics**

- Supported provider type for Bitbucket (p. 44)
- Supported provider type for GitHub and GitHub Enterprise Cloud (p. 44)
- Supported provider type and versions for GitHub Enterprise Server (p. 44)

### Supported provider type for Bitbucket

You can use the AWS CodeStar app with Atlassian Bitbucket Cloud.

### Supported provider type for GitHub and GitHub Enterprise Cloud

You can use the AWS Connector for GitHub app with GitHub and GitHub Enterprise Cloud.

### Supported provider type and versions for GitHub Enterprise Server

You can use the AWS CodeStar app with supported versions of GitHub Enterprise Server. For a list of supported versions, see [https://enterprise.github.com/releases/](https://enterprise.github.com/releases/).
Important
AWS CodeStar Connections does not support deprecated GitHub Enterprise Server versions. For example, AWS CodeStar Connections does not support GitHub Enterprise Server version 2.22.0 due to a known issue in the release. To connect, upgrade to version 2.22.1 or the latest available version.

Setting up connections
Complete the tasks in this section to get set up for creating and using the connections feature in the Developer Tools console.

Topics
- Sign up for AWS (p. 45)
- Create an IAM user with permissions to create connections (p. 45)

Sign up for AWS
If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account
2. Follow the online instructions.
   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

Create an IAM user with permissions to create connections
After you create an AWS account, you can use the AWS Management Console to create a user that has permissions to create connections.

To create an IAM user with permissions to create connections (console)
1. Use your AWS account email address and password to sign in as administrator to the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose Users, and then choose Add user.
3. For User name, enter the user name of your choice (for example, Connections_User).
4. Choose Next: Review. When you are ready to proceed, choose Create user.
5. Choose Download Credentials and save them in a location where you can easily retrieve them later.
6. Choose Close.
7. Choose the user name in the list.
8. Under Permissions, expand the Inline Policies header by choosing the down arrow on the right.
9. Where it says, There are no inline policies to show. To create one, click here, choose click here.
10. On the Set Permissions screen, choose Custom Policy.
11. Give your policy a name, such as ConnectionsUserPolicyConsole.
12. Choose Select.
13. Paste the following policy into Policy Document.

```json
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "codecommit:ListRepositories",
        "Resource": "*"
    }
}
```
Getting started with connections

The easiest way to get started with connections is to set up a connection that associates your third-party source repository to your AWS resources. If you wanted to connect your pipeline to an AWS source, such as CodeCommit, you would connect to it as a source action. However, if you have an external repository, you have to create a connection to associate your repository with your pipeline. In this tutorial, you set up a connection with your Bitbucket repository and your pipeline.

In this section, you use connections with:

- **AWS CodePipeline**: In these steps, you create a pipeline with your Bitbucket repository as the pipeline source.
- **Amazon CodeGuru Reviewer**: Next, you associate your Bitbucket repository to your feedback and analysis tools in CodeGuru Reviewer.

**Topics**

- Prerequisites (p. 46)
- Step 1: Edit your source file (p. 47)
- Step 2: Create your pipeline (p. 47)
- Step 3: Associate your repository with CodeGuru Reviewer (p. 50)

**Prerequisites**

Before you begin, complete the steps in Setting up (p. 9). You also need a third-party source repository that you want to connect to your AWS services and allow the connection to manage authentication for you. For example, you might want to connect a Bitbucket repository to your AWS services that integrate with source repositories.

- Create a Bitbucket repository with your Bitbucket account.
- Have your Bitbucket credentials ready. When you use the AWS Management Console to set up a connection, you are asked to sign in with your Bitbucket credentials.
Step 1: Edit your source file

When you create your Bitbucket repository, a default README.md file is included, which you will edit.

1. Log in to your Bitbucket repository and choose Source.
2. Choose the README.md file and choose Edit at the top of the page. Delete the existing text and add the following text.

   This is a Bitbucket repository!

3. Choose Commit.

   Make sure the README.md file is at the root level of your repository.

Step 2: Create your pipeline

In this section, you create a pipeline with the following actions:

- A source stage with a connection to your Bitbucket repository and action.
- A build stage with an AWS CodeBuild build action.

To create a pipeline with the wizard

2. On the Welcome page, Getting started page, or Pipelines page, choose Create pipeline.
3. In Step 1: Choose pipeline settings, in Pipeline name, enter MyBitbucketPipeline.
4. In Service role, choose Create service role.

   Note
   If you choose instead to use your existing CodePipeline service role, make sure that you have added the codestar-connections:UseConnection IAM permission to your service role policy. For instructions for the CodePipeline service role, see Add permissions to the the CodePipeline service role.

5. In Artifact store, choose Default location to use the default artifact store, such as the Amazon S3 artifact bucket designated as the default, for your pipeline in the Region you selected for your pipeline.

   Note
   This is not the source bucket for your source code. This is the artifact store for your pipeline. A separate artifact store, such as an S3 bucket, is required for each pipeline.

   Choose Next.
6. On the Step 2: Add source stage page, add a source stage:

   a. In Source provider, choose Bitbucket.
   b. Under Connection, choose Connect to Bitbucket.
   c. On the Connect to Bitbucket page, in Connection name, enter the name for the connection that you want to create. The name helps you identify this connection later.

   Under Bitbucket apps, choose Install a new app.
   d. On the app installation page, a message shows that the AWS CodeStar app is trying to connect to your Bitbucket account. Choose Grant access. After you have authorized the connection, your repositories on Bitbucket are detected, and you can choose to associate one with your AWS resource.
e. The connection ID for your new installation is displayed. Choose **Complete connection**.

f. In **Repository name**, choose the name of your Bitbucket repository.

g. In **Branch name**, choose the branch for your repository.

Choose **Next**.

7. In **Add build stage**, add a build stage:

a. In **Build provider**, choose **AWS CodeBuild**. Allow **Region** to default to the pipeline Region.

b. Choose **Create project**.

c. In **Project name**, enter a name for this build project.

d. In **Environment image**, choose **Managed image**. For **Operating system**, choose **Ubuntu**.

e. For **Runtime**, choose **Standard**. For **Image**, choose **aws/codebuild/standard:4.0**.

f. For **Service role**, choose **New service role**.

g. Under **Buildspec**, for **Build specifications**, choose **Insert build commands**. Choose **Switch to editor**, and paste the following under **Build commands**:

```plaintext
version: 0.2
phases:
  install:
    # If you use the Ubuntu standard image 2.0 or later, you must specify runtime-versions.
    # If you specify runtime-versions and use an image other than Ubuntu standard image 2.0, the build fails.
    runtime-versions:
      nodejs: 10
    # name: version
    #commands:
    # - command
    # - command
  pre_build:
    commands:
    - ls -lt
    - cat README.md
    # build:
    #commands:
    # - command
    # - command
  post_build:
    commands:
    # - command
    # - command
#artifacts:
#files:
  # - location
  # - location
#name: $(date +%Y-%m-%d)
#discard-paths: yes
#base-directory: location
#cache:
#paths:
# - paths
```

h. Choose **Continue to CodePipeline**. This returns to the CodePipeline console and creates a CodeBuild project that uses your build commands for configuration. The build project uses a service role to manage AWS service permissions. This step might take a couple of minutes.

i. Choose **Next**.
8. On the **Step 4: Add deploy stage** page, choose **Skip deploy stage**, and then accept the warning message by choosing **Skip** again. Choose **Next**.

9. On **Step 5: Review**, choose **Create pipeline**.

10. When your pipeline is successfully created, a pipeline execution starts.

11. On your successful build stage, choose **Details**.

   Under **Execution details**, view the CodeBuild build output. The commands output the README.md file contents as follows:

   ```
   This is a Bitbucket repository!
   ```
Step 3: Associate your repository with CodeGuru Reviewer

After you create a connection, you can use that connection for all of your AWS resources in the same account. For example, you can use the same Bitbucket connection for a CodePipeline source action in a pipeline and your repository commit analysis in CodeGuru Reviewer.

1. Sign in to the CodeGuru Reviewer console.
2. Under CodeGuru Reviewer, choose Associate repository.

   The one-page wizard opens.
3. Under Select source provider, choose Bitbucket.
4. Under Connect to Bitbucket (with AWS CodeStar connections), choose the connection you created for your pipeline.
5. Under Repository location, choose the name of your Bitbucket repository, and choose Associate.

You can continue to set up code reviews. For more information, see Connecting to Bitbucket to associate a repository with CodeGuru Reviewer in the Amazon CodeGuru Reviewer User Guide.

Working with connections

Connections are configurations that you use to connect AWS resources to external code repositories. Each connection is a resource that can be given to services such as AWS CodePipeline to connect to a third-party repository such as Bitbucket. For example, you can add the connection in CodePipeline so that it triggers your pipeline when a code change is made to your third-party code repository. You can also connect your AWS resources to an installed provider type such as GitHub Enterprise Server.

If you want to create a connection to an installed provider type, such as GitHub Enterprise Server, the console creates a host for you. A host is a resource that you create to represent the server where your provider is installed. For more information, see Working with hosts (p. 77).

When you create a connection, you use a wizard in the console to install the AWS CodeStar app with your third-party provider and associate it with a new connection. If you have already installed the AWS CodeStar app, you can use it.

For more information about connections, see the AWS CodeStar Connections API reference. For more information about the CodePipeline source action for Bitbucket, see CodestarConnectionSource in the AWS CodePipeline User Guide.

To create or attach a policy to your AWS Identity and Access Management (IAM) user or role with the permissions required to use AWS CodeStar connections, see AWS CodeStar Connections permissions reference (p. 105). Depending on when your CodePipeline service role was created, you might need to update its permissions to support AWS CodeStar connections. For instructions, see Update the service role in the AWS CodePipeline User Guide.

Topics
- Create a connection (p. 51)
- Create a connection to Bitbucket (p. 51)
- Create a connection to GitHub (p. 54)
- Create a connection to GitHub Enterprise Server (p. 59)
- Update a pending connection (p. 66)
- List connections (p. 68)
- Delete a connection (p. 69)
Create a connection

You can create connections to the following third-party provider types:

- To create a connection to Bitbucket, see Create a connection to Bitbucket (p. 51).
- To create a connection to GitHub or GitHub Enterprise Cloud, see Create a connection to GitHub (p. 54).
- To create a connection to GitHub Enterprise Server, including creating your host resource, see Create a connection to GitHub Enterprise Server (p. 59).

**Important**

AWS CodeStar Connections does not support GitHub Enterprise Server version 2.22.0 due to a known issue in the release. To connect, upgrade to version 2.22.1 or the latest available version.

Create a connection to Bitbucket

You can use the AWS Management Console or the AWS Command Line Interface (AWS CLI) to create a connection to a repository hosted on bitbucket.org.

Before you begin:

- You must have already created an account with Bitbucket.
- You must have already created a code repository on bitbucket.org.

**Note**

You can create connections to a Bitbucket Cloud repository. Installed Bitbucket provider types, such as Bitbucket Server, are not supported. See AWS CodeStar Connections supported providers and versions (p. 44).

**Note**

Connections only provide access to repositories owned by the account that was used to create the connection.

**Topics**

- Create a connection to Bitbucket (console) (p. 51)
- Create a connection to Bitbucket (CLI) (p. 54)

Create a connection to Bitbucket (console)

**Step 1: Create your connection**

2. Choose **Settings > Connections**, and then choose **Create connection**.
3. To create a connection to a Bitbucket repository, under **Select a provider**, choose **Bitbucket**. In **Connection name**, enter the name for the connection that you want to create. Choose **Connect to Bitbucket**, and proceed to Step 2.
Step 2: Connect to Bitbucket

1. On the Connect to Bitbucket settings page, your connection name displays.
   
   Under Bitbucket apps, choose an app installation or choose Install a new app to create one.

   **Note**
   You install one app for all of your connections to a particular provider. If you have already installed the Bitbucket app, choose it and move to the last step in this section.

2. If the login page for Bitbucket displays, log in with your credentials and then choose to continue.
3. On the app installation page, a message shows that the AWS CodeStar app is trying to connect to your Bitbucket account. Choose **Grant access**.

![AWS CodeStar requests access](image)

**AWS CodeStar requests access**
This app is hosted at https://codestar-connections.webhooks.aws

Read your account information
Read your repositories and their pull requests
Administer your repositories
Read and modify your repositories

Authorize for

![Authorization button](image)

**Allow AWS CodeStar to do this?**
This 3rd party vendor has not provided a privacy policy or terms of use.
Atlassian's Privacy Policy is not applicable to the use of this App.

![Grant access button](image)

4. In **Bitbucket apps**, the connection ID for your new installation is displayed. Choose **Connect**. The created connection displays in the connections list.
Create a connection to Bitbucket (CLI)

You can use the AWS Command Line Interface (AWS CLI) to create a connection.

To do this, use the `create-connection` command.

**Important**
A connection created through the AWS CLI or AWS CloudFormation is in `PENDING` status by default. After you create a connection with the CLI or AWS CloudFormation, use the console to edit the connection to make its status `AVAILABLE`.

**To create a connection to Bitbucket**

1. Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the `create-connection` command, specifying the `--provider-type` and `--connection-name` for your connection. In this example, the third-party provider name is `Bitbucket` and the specified connection name is `MyConnection`.

   ```bash
   aws codestar-connections create-connection --provider-type Bitbucket --connection-name MyConnection
   ```

   If successful, this command returns the connection ARN information similar to the following.

   ```json
   {
     "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
   }
   ```

2. Use the console to complete the connection. For more information, see Update a pending connection (p. 66).

Create a connection to GitHub

You can use the AWS Management Console or the AWS Command Line Interface (AWS CLI) to create a connection to GitHub.

Before you begin:

- You must have already created an account with GitHub.
- You must have already created your third-party code repository.

**Note**
Connections only provide access to repositories owned by the account that was used to create the connection.

**Topics**
- Create a connection to GitHub (console) (p. 54)
- Create a connection to GitHub (CLI) (p. 59)

Create a connection to GitHub (console)

2. Choose **Settings > Connections**, and then choose **Create connection**.

3. To create a connection to a GitHub or GitHub Enterprise Cloud repository, under **Select a provider**, choose **GitHub**. In **Connection name**, enter the name for the connection that you want to create. Choose **Connect to GitHub**, and proceed to Step 2.

---

**Create a connection**

**Select a provider**

- Bitbucket
- GitHub
- GitHub Enterprise Server

**Create GitHub App connection**

**Connection name**

- github-conection

**Connect to GitHub**

---

### To create a connection to GitHub

1. Under **GitHub connection settings**, your connection name appears in **Connection name**. Choose **Connect to GitHub**. The access request page appears.
2. Choose **Authorize AWS Connector for GitHub**. The connection page displays and shows the **GitHub Apps** field.

3. Under **GitHub Apps**, choose an app installation or choose **Install a new app** to create one.
Note
You install one app for all of your connections to a particular provider. If you have already installed the AWS Connector for GitHub app, choose it and skip this step.

4. On the Install AWS Connector for GitHub page, choose the account where you want to install the app.

Note
You only install the app once for each GitHub account. If you previously installed the app, you can choose Configure to proceed to a modification page for your app installation, or you can use the back button to return to the console.

5. On the Install AWS Connector for GitHub page, leave the defaults, and choose Install.
6. On the **Connect to GitHub** page, the connection ID for your new installation appears in **GitHub Apps**. Choose **Connect**.

**View your created connection**

- The created connection displays in the connections list.
Create a connection to GitHub (CLI)

You can use the AWS Command Line Interface (AWS CLI) to create a connection to GitHub.

To do this, use the `create-connection` command.

**Important**
A connection created through the AWS CLI or AWS CloudFormation is in PENDING status by default. After you create a connection with the CLI or AWS CloudFormation, use the console to edit the connection to make its status AVAILABLE.

**To create a connection to GitHub**

1. Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the `create-connection` command, specifying the `--provider-type` and `--connection-name` for your connection. In this example, the third-party provider name is GitHub and the specified connection name is `MyConnection`.

   ```bash
   aws codestar-connections create-connection --provider-type GitHub --connection-name MyConnection
   ```

   If successful, this command returns the connection ARN information similar to the following.

   ```json
   {
     "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfca80bc441f"
   }
   ```

2. Use the console to complete the connection. For more information, see Update a pending connection (p. 66).

Create a connection to GitHub Enterprise Server

You use connections to associate your AWS resources with a third-party repository. You can use the AWS Management Console or the AWS Command Line Interface (AWS CLI) to create a connection to GitHub Enterprise Server.

Connections only provide access to repositories owned by the GitHub Enterprise Server account that is used during connection creation to authorize installation of the GitHub app.

Before you begin:

- You must already have a GitHub Enterprise Server instance and a repository in it.
- You need to be an administrator of the GitHub Enterprise Server instance in order to create GitHub apps and create a host resource as shown in this section.

**Important**
When you set up your host for GitHub Enterprise Server, a VPC endpoint for webhooks event data is created for you. If you created your host before November 24, 2020, and you want to use VPC PrivateLink webhook endpoints, you must first delete your host and then create a new host.

**Topics**

- Create a connection to GitHub Enterprise Server (console) (p. 60)
- Create a connection to GitHub Enterprise Server (CLI) (p. 65)
Create a connection to GitHub Enterprise Server (console)

To create a GitHub Enterprise Server connection, you provide information for where your GitHub Enterprise Server is installed and authorize the connection creation with your GitHub Enterprise credentials.

**Topics**

- Prerequisites: Network or Amazon VPC configuration for your connection (p. 60)
- Create your GitHub Enterprise Server connection (console) (p. 61)

**Prerequisites: Network or Amazon VPC configuration for your connection**

If your infrastructure is configured with a network connection, you can skip this section and continue to Create your GitHub Enterprise Server connection (console) (p. 61).

If your GitHub Enterprise Server is only accessible in a VPC, follow these VPC requirements before you continue to Create your GitHub Enterprise Server connection (console) (p. 61).

**Important**

AWS CodeStar Connections does not support GitHub Enterprise Server version 2.22.0 due to a known issue in the release. To connect, upgrade to version 2.22.1 or the latest available version.

**VPC requirements**

The following are general VPC requirements, depending on the VPC you have set up for your installation.

- You can configure a *public* VPC with public and private subnets. You can use the default VPC for your AWS account if you do not have preferred CIDR blocks or subnets.
- If you have a *private* VPC configured, and you have configured your GitHub Enterprise Server instance to perform TLS validation using a non-public certificate authority, you need to provide the TLS certificate for your host resource.
- When AWS CodeStar Connections creates your host, the VPC endpoint (PrivateLink) for webhooks is created for you. For more information, see AWS CodeStar Connections and interface VPC endpoints (AWS PrivateLink) (p. 86).
- Security group configuration:
  - The security groups used during host creation need inbound and outbound rules that allow the network interface to connect to your GitHub Enterprise Server instance
  - The security groups attached to your GitHub Enterprise Server instance (not part of the host setup) need inbound and outbound access from the network interfaces created by connections.
- Your VPC subnets must reside in different Availability Zones in your Region. Availability Zones are distinct locations that are isolated from failures in other Availability Zones. Each subnet must reside entirely within one Availability Zone and cannot span zones.

For more information about working with VPCs and subnets, see VPC and Subnet Sizing for IPv4 in the Amazon VPC User Guide.

**VPC information you provide for host setup**

When you create your host resource for your connections in the next step, you need to provide the following:

- **VPC ID:** The ID of the VPC for the server where your GitHub Enterprise Server instance is installed or a VPC which has access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.
• **Subnet ID or IDs:** The ID of the subnet for the server where your GitHub Enterprise Server instance is installed or a subnet with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.

• **Security group or groups:** The security group for the server where your GitHub Enterprise Server instance is installed or a security group with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.

• **Endpoint:** Have your server endpoint ready and continue to the next step.

For more information, including troubleshooting VPC or host connections, see Troubleshooting VPC configuration for your host (p. 91).

**Permission requirements**

As part of the host creation process, AWS CodeStar Connections creates network resources on your behalf to facilitate the VPC connectivity. This includes a network interface for AWS CodeStar Connections to query data from your host, and a VPC endpoint or PrivateLink for the host to send event data via webhooks to AWS CodeStar Connections. To be able to create these network resources, you must ensure the IAM user creating the host has the following permissions:

```plaintext
ec2:CreateNetworkInterface
ec2:CreateTags
ec2:DescribeDhcpOptions
ec2:DescribeNetworkInterfaces
ec2:DescribeSubnets
ec2:DeleteNetworkInterface
ec2:CreateVpcEndpoint
ec2:DeleteVpcEndpoints
ec2:DescribeVpcEndpoints
```

For more information about troubleshooting permissions or host connections in a VPC, see Troubleshooting VPC configuration for your host (p. 91).

For more information about the webhook VPC endpoint, see AWS CodeStar Connections and interface VPC endpoints (AWS PrivateLink) (p. 86).

**Create your GitHub Enterprise Server connection (console)**

To create a connection to GitHub Enterprise Server, have your server URL and GitHub Enterprise credentials ready.

**Step 1: Create your connection**

2. Choose **Settings > Connections**, and then choose **Create connection**.
3. To create a connection to an installed GitHub Enterprise Server repository, choose **GitHub Enterprise Server**.

**Connect to GitHub Enterprise Server**

1. In **Connection name**, enter the name for your connection.
2. In **URL**, enter the endpoint for your server.

   **Note**
   If the provided URL has already been used to set up a GitHub Enterprise Server for a connection, you will be prompted to choose the host resource ARN that was created previously for that endpoint.

3. If you have launched your server into an Amazon VPC and you want to connect with your VPC, choose **Use a VPC** and complete the following.
   - In **VPC ID**, choose your VPC ID. Make sure to choose the VPC for the infrastructure where your GitHub Enterprise Server instance is installed or a VPC with access to your GitHub Enterprise Server instance through VPN or Direct Connect.
   - Under **Subnet ID**, choose **Add**. In the field, choose the subnet ID you want to use for your host. You can choose up to 10 subnets.

     Make sure to choose the subnet for the infrastructure where your GitHub Enterprise Server instance is installed or a subnet with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.

   - Under **Security group IDs**, choose **Add**. In the field, choose the security group you want to use for your host. You can choose up to 10 security groups.

     Make sure to choose the security group for the infrastructure where your GitHub Enterprise Server instance is installed or a security group with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.
d. If you have a private VPC configured, and you have configured your GitHub Enterprise Server instance to perform TLS validation using a non-public certificate authority, in **TLS certificate**, enter your certificate ID. The TLS Certificate value should be the public key of the certificate.

![VPC ID](image)

**VPC ID**
Choose the VPC in which your GitHub Enterprise Server is configured.

![Subnet IDs](image)

**Subnet IDs**
Choose the subnet or subnets for the VPC in which your GitHub Enterprise Server is configured.

![Security group IDs](image)

**Security group IDs**
Choose the security group or groups for the VPC in which your GitHub Enterprise Server is configured.

**TLS certificate - optional**
If you have a private certificate authority behind a VPC or you are using a self-signed certificate paste the TLS certificate here.

4. Choose **Connect to GitHub Enterprise Server**. The created connection is shown with a **Pending** status. A host resource is created for the connection with the server information you provided. For the host name, the URL is used.

5. Choose **Update pending connection**.
6. If prompted, on the GitHub Enterprise login page, sign in with your GitHub Enterprise credentials.
7. On the Create GitHub App page, choose a name for your app.

8. On the GitHub authorization page, choose **Authorize <app-name>**.
9. On the app installation page, a message shows that the AWS CodeStar Connector app is ready to be installed. If you have multiple organizations, you might be prompted to choose the organization where you want to install the app.

Choose the repository settings where you want to install the app. Choose Install.

10. The connection page shows the created connection in an Available status.

Create a connection to GitHub Enterprise Server (CLI)

You can use the AWS Command Line Interface (AWS CLI) to create a connection.

To do this, use the create-connection command.
Important

A connection created through the AWS CLI or AWS CloudFormation is in PENDING status by default. After you create a connection with the CLI or AWS CloudFormation, use the console to edit the connection to make its status AVAILABLE.

To create a connection for an installed provider type

1. Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the `create-connection` command, specifying the `--host-arn` and `--connection-name` for your connection.

   ```bash
   aws codestar-connections create-connection --host-arn arn:aws:codestar-connections:us-west-2:account_id:host/MyHost-234EXAMPLE --connection-name MyConnection
   ```

   If successful, this command returns the connection ARN information similar to the following.

   ```json
   {
      "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad"
   }
   ```

2. Use the console to set up the pending connection. For more information, see Update a pending connection (p. 66).

Update a pending connection

A connection created through the AWS Command Line Interface (AWS CLI) or AWS CloudFormation is in PENDING status by default. After you create a connection with the AWS CLI or AWS CloudFormation, use the console to update the connection to make its status AVAILABLE.

Note

You must use the console to update a pending connection. You cannot update a pending connection using the AWS CLI.

The first time you use the console to add a new connection to a third-party provider, you must complete the OAuth handshake with the third-party provider using the installation associated with your connection.

You can use the Developer Tools console to complete a pending connection.

To complete a connection


2. Choose Settings > Connections.

   The names of all connections associated with your AWS account are displayed.

3. In Name, choose the name of the pending connection you want to update.

   **Update a pending connection** is enabled when you choose a connection with a Pending status.

4. Choose Update a pending connection.

5. On the Connect to Bitbucket page, in **Connection name**, verify the name of your connection.

   Under Bitbucket apps, choose an app installation, or choose **Install a new app** to create one.
6. On the app installation page, a message shows that the AWS CodeStar app is trying to connect to your Bitbucket account. Choose **Grant access**.

7. The connection ID for your new installation is displayed. Choose **Complete connection**.
List connections

You can use the Developer Tools console or the `list-connections` command in the AWS Command Line Interface (AWS CLI) to view a list of connections in your account.

List connections (console)

To list connections

2. Choose Settings > Connections.
3. View the name, status, and ARN for your connections.

List connections (CLI)

You can use the AWS CLI to list your connections to third-party code repositories. For a connection associated to a host resource, such as connections to GitHub Enterprise Server, the output additionally returns the host ARN.

To do this, use the `list-connections` command.

To list connections

- Open a terminal (Linux, macOS, or Unix) or command prompt (Windows), and use the AWS CLI to run the `list-connections` command.

```plaintext
aws codestar-connections list-connections --provider-type Bitbucket --max-results 5 --next-token: next-token
```

This command returns the following output.

```json
{
  "Connections": [
    {
      "ConnectionName": "my-connection",
      "ProviderType": "Bitbucket",
      "Status": "PENDING",
      "ARN": "arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfca0bc441f",
      "OwnerAccountId": "account_id"
    },
    {
      "ConnectionName": "my-other-connection",
      "ProviderType": "Bitbucket",
      "Status": "AVAILABLE",
      "ARN": "arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfca0bc441f",
      "OwnerAccountId": "account_id"
    }
  ],
  "NextToken": "next-token"
}
```
Delete a connection

You can use the Developer Tools console or the delete-connection command in the AWS Command Line Interface (AWS CLI) to delete a connection.

Topics
- Delete a connection (console) (p. 69)
- Delete a connection (CLI) (p. 69)

Delete a connection (console)

To delete a connection

2. Choose Settings > Connections.
3. In Connection name, choose the name of the connection you want to delete.
4. Choose Delete.
5. Enter delete in the field to confirm, and then choose Delete.

Important
This action cannot be undone.

Delete a connection (CLI)

You can use the AWS Command Line Interface (AWS CLI) to delete a connection.

To do this, use the delete-connection command.

Important
After you run the command, the connection is deleted. No confirmation dialog box is displayed. You can create a new connection, but the Amazon Resource Name (ARN) is never reused.

To delete a connection

- Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the delete-connection command, specifying the ARN of the connection that you want to delete.

    aws codestar-connections delete-connection --connection-arn arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f

This command returns nothing.

Tag connections resources

A tag is a custom attribute label that you or AWS assigns to an AWS resource. Each AWS tag has two parts:

- A tag key (for example, CostCenter, Environment, or Project). Tag keys are case sensitive.
- An optional field known as a tag value (for example, 111122223333, Production, or a team name). Omitting the tag value is the same as using an empty string. Like tag keys, tag values are case sensitive.
Together these are known as *key-value pairs*.

You can use the console or the CLI to tag resources.

You can tag the following resource types in AWS CodeStar connections:

- Connections
- Hosts

These steps assume that you have already installed a recent version of the AWS CLI or updated to the current version. For more information, see *Installing the AWS CLI* in the *AWS Command Line Interface User Guide*.

In addition to identifying, organizing, and tracking your resource with tags, you can use tags in AWS Identity and Access Management (IAM) policies to help control who can view and interact with your resource. For examples of tag-based access policies, see *Using tags to control access to AWS CodeStar Connections resources* (p. 124).

**Topics**

- Tag resources (console) (p. 70)
- Tag resources (CLI) (p. 73)

### Tag resources (console)

You can use the console to add, update, or remove tags on a connections resource.

**Topics**

- Add tags to a connections resource (console) (p. 70)
- View tags for a connections resource (console) (p. 71)
- Edit tags for a connections resource (console) (p. 72)
- Remove tags from a connections resource (console) (p. 72)

### Add tags to a connections resource (console)

You can use the console to add tags to an existing connection or host.

**Note**

When you create a connection for an installed provider such as GitHub Enterprise Server, and a host resource is also created for you, the tags during creation are added to the connection only. This allows you to tag a host separately if you want to reuse it for a new connection. If you want to add tags to the host, use the steps here.

**To add tags for a connection**

1. Sign in to the console. From the navigation pane, choose *Settings*.
2. Under *Settings*, choose *Connections*. Choose the *Connections* tab.
3. Choose the connection you want to edit. The connection settings page displays.
4. Under *Connection tags*, choose *Edit*. The *Edit Connection tags* page displays.
5. In the *Key* and *Value* fields, enter a key pair for each set of tags you want to add. (The *Value* field is optional.) For example, in *Key*, enter *Project*. In *Value*, enter *ProjectA*. 
6. (Optional) Choose **Add tag** to add more rows and enter more tags.
7. Choose **Submit**. The tags are listed under connection settings.

**To add tags for a host**

1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Hosts** tab.
3. Choose the host you want to edit. The host settings page displays.
4. Under **Host tags**, choose **Edit**. The **Host tags** page displays.
5. In the **Key** and **Value** fields, enter a key pair for each set of tags you want to add. (The **Value** field is optional.) For example, in **Key**, enter **Project**. In **Value**, enter **ProjectA**.

6. (Optional) Choose **Add tag** to add more rows and enter more tags for a host.
7. Choose **Submit**. The tags are listed under host settings.

**View tags for a connections resource (console)**

You can use the console to view the tags for existing resources.

**To view tags for a connection**

1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Connections** tab.
3. Choose the connection you want to view. The connection settings page displays.
4. Under **Connection tags**, view the tags for the connection under the **Key** and **Value** columns.

**To view tags for a host**
1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Hosts** tab.
3. Choose the host you want to view.
4. Under **Host tags**, view the tags for the host under the **Key** and **Value** columns.

**Edit tags for a connections resource (console)**
You can use the console to edit tags that have been added to connections resources.

**To edit tags for a connection**
1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Connections** tab.
3. Choose the connection you want to edit. The connection settings page displays.
4. Under **Connection tags**, choose **Edit**. The **Connection tags** page displays.
5. In the **Key** and **Value** fields, update the values in each field as needed. For example, for the **Project** key, in **Value**, change **ProjectA** to **ProjectB**.
6. Choose **Submit**.

**To edit tags for a host**
1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Hosts** tab.
3. Choose the host you want to edit. The host settings page displays.
4. Under **Host tags**, choose **Edit**. The **Host tags** page displays.
5. In the **Key** and **Value** fields, update the values in each field as needed. For example, for the **Project** key, in **Value**, change **ProjectA** to **ProjectB**.
6. Choose **Submit**.

**Remove tags from a connections resource (console)**
You can use the console to remove tags from connections resources. When you remove tags from the associated resource, the tags are deleted.

**To remove tags for a connection**
1. Sign in to the console. From the navigation pane, choose **Settings**.
2. Under **Settings**, choose **Connections**. Choose the **Connections** tab.
3. Choose the connection you want to edit. The connection settings page displays.
4. Under **Connection tags**, choose **Edit**. The **Connection tags** page displays.
5. Next to the key and value for each tag you want to delete, choose **Remove tag**.
6. Choose **Submit**.

**To remove tags for a host**
1. Sign in to the console. From the navigation pane, choose Settings.
3. Choose the host you want to edit. The host settings page displays.
4. Under Host tags, choose Edit. The Host tags page displays.
5. Next to the key and value for each tag you want to delete, choose Remove tag.
6. Choose Submit.

Tag resources (CLI)

You can use the CLI to view, add, update, or remove tags on a connections resource.

Topics
- Add tags to a connections resource (CLI) (p. 73)
- View tags for a connections resource (CLI) (p. 74)
- Edit tags for a connections resource (CLI) (p. 75)
- Remove tags from a connections resource (CLI) (p. 75)

Add tags to a connections resource (CLI)

You can use the AWS CLI to tag resources in connections.

At the terminal or command line, run the `tag-resource` command, specifying the Amazon Resource Name (ARN) of the resource where you want to add tags and the key and value of the tag you want to add. You can add more than one tag.

To add tags for a connection

1. Get the ARN for your resource. Use the `list-connections` command shown in List connections (p. 68) to get the connection ARN.
2. In a terminal or at the command line, run the `tag-resource` command.

For example, use the following command to tag a connection with two tags, a tag key named `Project` with the tag value of `ProjectA`, and a tag key named `ReadOnly` with the tag value of `true`.

```
Key=Project,Value=ProjectA
Key=IscontainerBased,Value=true
```

If successful, this command returns nothing.

To add tags for a host

1. Get the ARN for your resource. Use the `list-hosts` command shown in List hosts (p. 81) to get the host ARN.
2. In a terminal or at the command line, run the `tag-resource` command.

For example, use the following command to tag a host with two tags, a tag key named `Project` with the tag value of `ProjectA`, and a tag key named `IscontainerBased` with the tag value of `true`.

```
Key=Project,Value=ProjectA
Key=IscontainerBased,Value=true
```
   Key=IscontainerBased,Value=true

If successful, this command returns nothing.

**View tags for a connections resource (CLI)**

You can use the AWS CLI to view the AWS tags for a connections resource. If no tags have been added, the returned list is empty. Use the `list-tags-for-resource` command to view tags that have been added to a connection or a host.

**To view tags for a connection**

1. Get the ARN for your resource. Use the `list-connections` command shown in List connections (p. 68) to get the connection ARN.
2. In a terminal or at the command line, run the `list-tags-for-resource` command. For example, use the following command to view a list of tag keys and tag values for a connection.

   ```bash
   aws codestar-connections list-tags-for-resource --resource-arn arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
   ```

   This command returns the tags associated with the resource. This example shows two key-value pairs returned for a connection.

   ```json
   {
      "Tags": [
         {
            "Key": "Project",
            "Value": "ProjectA"
         },
         {
            "Key": "ReadOnly",
            "Value": "true"
         }
      ]
   }
   ```

**To view tags for a host**

1. Get the ARN for your resource. Use the `list-hosts` command shown in List hosts (p. 81) to get the host ARN.
2. In a terminal or at the command line, run the `list-tags-for-resource` command. For example, use the following command to view a list of tag keys and tag values for a host.

   ```bash
   ```

   This command returns the tags associated with the resource. This example shows two key-value pairs returned for a host.

   ```json
   {
      "Tags": [
         {
```
Edit tags for a connections resource (CLI)

You can use the AWS CLI to edit a tag for a resource. You can change the value for an existing key or add another key.

At the terminal or command line, run the `tag-resource` command, specifying the ARN of the resource where you want to update a tag and specify the tag key and tag value to update.

When you edit tags, any tag keys not specified will be retained, while anything with the same key but a new value will be updated. New keys that are added with the edit command are added as a new key-value pair.

**To edit tags for a connection**

1. Get the ARN for your resource. Use the `list-connections` command shown in List connections (p. 68) to get the connection ARN.
2. In a terminal or at the command line, run the `tag-resource` command.

   In this example, the value for the key `Project` is changed to `ProjectB`.

   ```bash
   ```

   If successful, this command returns nothing. To verify the tags associated with the connection, run the `list-tags-for-resource` command.

**To edit tags for a host**

1. Get the ARN for your resource. Use the `list-hosts` command shown in List hosts (p. 81) to get the host ARN.
2. In a terminal or at the command line, run the `tag-resource` command.

   In this example, the value for the key `Project` is changed to `ProjectB`.

   ```bash
   ```

   If successful, this command returns nothing. To verify the tags associated with the host, run the `list-tags-for-resource` command.

Remove tags from a connections resource (CLI)

Follow these steps to use the AWS CLI to remove a tag from a resource. When you remove tags from the associated resource, the tags are deleted.
**Note**
If you delete a connection resource, all tag associations are removed from the deleted resource.
You do not have to remove tags before you delete a connection resource.

At the terminal or command line, run the `untag-resource` command, specifying the ARN of the resource where you want to remove tags and the tag key of the tag you want to remove. For example, to remove multiple tags on a connection with the tag keys `Project` and `ReadOnly`, use the following command.

```bash
```

If successful, this command returns nothing. To verify the tags associated with the resource, run the `list-tags-for-resource` command. The output shows that all tags have been removed.

```
{
  "Tags": []
}
```

### View connection details

You can use the Developer Tools console or the `get-connection` command in the AWS Command Line Interface (AWS CLI) to view details for a connection. To use the AWS CLI, you must have already installed a recent version of the AWS CLI or updated to the current version. For more information, see Installing the AWS CLI in the AWS Command Line Interface User Guide.

**To view a connection (console)**

2. Choose Settings > Connections.
3. Choose the button next to the connection you want to view, and then choose View details.
4. The following information appears for your connection:
   - The connection name.
   - The provider type for your connection.
   - The connection status.
   - The connection ARN.
   - If the connection was created for an installed provider, such as GitHub Enterprise Server, the host information associated with the connection.
   - If the connection was created for an installed provider, such as GitHub Enterprise Server, the endpoint information associated with the host for the connection.
5. If the connection is in Pending status, to complete the connection, choose Update pending connection. For more information, see Update a pending connection (p. 66).

**To view a connection (CLI)**

- At the terminal or command line, run the `get-connection` command. For example, use the following command to view details for a connection with the `arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f` ARN value.

```bash
aws codestar-connections get-connection --connection-arn arn:aws:codestar-connections:us-west-2:account_id:connection/aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f
```
If successful, this command returns the connections details.

Example output for a Bitbucket connection:

```json
{
    "Connection": {
        "ConnectionName": "MyConnection",
        "ProviderType": "Bitbucket",
        "OwnerAccountId": "account_id",
        "ConnectionStatus": "AVAILABLE"
    }
}
```

Example output for a GitHub connection:

```json
{
    "Connection": {
        "ConnectionName": "MyGitHubConnection",
        "ProviderType": "GitHub",
        "OwnerAccountId": "account_id",
        "ConnectionStatus": "AVAILABLE"
    }
}
```

Example output for a GitHub Enterprise Server connection:

```json
{
    "Connection": {
        "ConnectionName": "MyConnection",
        "ProviderType": "GitHubEnterpriseServer",
        "OwnerAccountId": "account_id",
        "ConnectionStatus": "PENDING",
        "HostArn": "arn:aws:codestar-connections:us-west-2:account_id:host/sdfasdf-EXAMPLE"
    }
}
```

**Working with hosts**

To create a connection to an installed provider type, such as GitHub Enterprise Server, you first create a host using the AWS Management Console. A host is a resource that you create to represent the infrastructure where your provider is installed. Then you create a connection using that host. For more information, see Working with connections (p. 50).

For example, you create a host for your connection so that the third-party app for your provider can be registered to represent your infrastructure. You create one host for a provider type, and then all of your connections to that provider type use that host.

When you use the console to create a connection to an installed provider type, such as GitHub Enterprise Server, the console creates your host resource for you.
Create a host

You can use the AWS Management Console or the AWS Command Line Interface (AWS CLI) to create a connection to a third-party code repository that is installed on your infrastructure. For example, you might have GitHub Enterprise Server running as a virtual machine on an Amazon EC2 instance. Before you create a connection to GitHub Enterprise Server, you create a host to use for the connection.

Before you begin:

- You must have already created a network or virtual private cloud (VPC).
- You must have already created your instance and, if you plan to connect with your VPC, launched your instance into your VPC.

For more information about network configuration for your host resource, see Troubleshooting VPC configuration for your host (p. 91).

To use the console to create a connection to GitHub Enterprise Server, see Create your GitHub Enterprise Server connection (console) (p. 61). The console creates your host for you.

Important
AWS CodeStar Connections does not support GitHub Enterprise Server version 2.22.0 due to a known issue in the release. For more information, see the release notes Known Issues list at https://enterprise.github.com/releases/series/2.22.

Topics
- Create a host for a connection (console) (p. 78)
- Create a host (CLI) (p. 79)
5. In **URL**, enter the endpoint for the infrastructure where your provider is installed.
6. If your server is configured within an Amazon VPC and you want to connect with your VPC, choose **Use a VPC**. Otherwise, choose **No VPC**.
7. If you have launched your instance into an Amazon VPC and you want to connect with your VPC, choose **Use a VPC** and complete the following.
   a. In **VPC ID**, choose your VPC ID. Make sure to choose the VPC for the infrastructure where your GitHub Enterprise Server instance is installed or a VPC with access to your GitHub Enterprise Server instance through VPN or Direct Connect.
   b. If you have a private VPC configured, and you have configured your GitHub Enterprise Server instance to perform TLS validation using a non-public certificate authority, in **TLS certificate**, enter your certificate ID. The TLS Certificate value is the public key of the certificate.
8. Choose **Create host**.
9. After the host details page displays, the host status changes as the host is created.  
   **Note**
   If your host setup includes a VPC configuration, allow several minutes for provisioning of host network components.

Wait for your host to reach a **Pending** status, and then complete the setup. For more information, see [Set up a pending host](#) (p. 80).

### Create a host (CLI)

You can use the AWS Command Line Interface (AWS CLI) to create a host for installed connections.

**Note**
You only create a host once per GitHub Enterprise Server account. All of your connections to a specific GitHub Enterprise Server account will use the same host.

You use a host to represent the endpoint for the infrastructure where your third-party provider is installed. To create a host with the CLI, you use the **create-host** command. After you finish creating the host, the host is in **Pending** status. You then **set up** the host to move it to an **Available** status. After the host is available, you complete the steps to create a connection.

**Important**
A host created through the AWS CLI is in **Pending** status by default. After you create a host with the CLI, use the console to set up the host to make its status **Available**.

**To create a host**

1. Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the **create-host** command, specifying the --name, --provider-type, and --provider-endpoint parameters.
for your connection. In this example, the third-party provider name is GitHubEnterpriseServer and the endpoint is my-instance.dev.

```bash/aws codestar-connections create-host --name MyHost --provider-type GitHubEnterpriseServer --provider-endpoint "https://my-instance.dev"
```

If successful, this command returns the host Amazon Resource Name (ARN) information similar to the following.

```
{
    "HostArn": "arn:aws:codestar-connections:us-west-2:account_id:host/MyHost-28aef605"
}
```

After this step, the host is in **PENDING** status.

2. Use the console to complete the host setup and move the host to an **Available** status. For more information, see Set up a pending host (p. 80).

### Set up a pending host

A host created through the AWS Command Line Interface (AWS CLI) or SDK is in **Pending** status by default. After you create a connection with the console, AWS CLI, or the SDK, use the console to set up the host to make its status **Available**.

You must have already created a host. For more information, see Create a host (p. 78).

**To set up a pending host**

After your host is created, it is in a **Pending** status. To move the host from **Pending** to **Available**, complete these steps. This process performs a handshake with the third-party provider to register the AWS connection app on the host.

1. After your host reaches **Pending** status on the AWS Developer Tools console, choose **Set up host**.
2. On the third-party installed provider login page, such as the **GitHub Enterprise Server** login page, log in with your account credentials if prompted.
3. On the app install page, in **GitHub App name**, enter a name for the app you want to install for your host. Choose **Create GitHub App**.
4. After your host is successfully registered, the host details page appears and shows that the host status is **Available**.
5. You can continue with creating your connection after the host is available. On the success banner, choose **Create connection**. Complete the steps in Create a connection (p. 60).
**List hosts**

You can use the Developer Tools console or the `list-connections` command in the AWS Command Line Interface (AWS CLI) to view a list of connections in your account.

**List hosts (console)**

**To list hosts**

2. Choose the **Hosts** tab. View the name, status, and ARN for your hosts.

**List hosts (CLI)**

You can use the AWS CLI to list your hosts for installed third-party provider connections.

To do this, use the `list-hosts` command.

**To list hosts**

- Open a terminal (Linux, macOS, or Unix) or command prompt (Windows), and use the AWS CLI to run the `list-hosts` command.

```bash
aws codestar-connections list-hosts
```

This command returns the following output.

```json
{
  "Hosts": [
    {
      "Name": "My-Host",
      "ProviderType": "GitHubEnterpriseServer",
      "ProviderEndpoint": "https://my-instance.test.dev",
      "Status": "AVAILABLE"
    }
  ]
}
```

**Edit a host**

You can edit host settings for a host in Pending status. You can edit the host name, URL, or VPC configuration.

You cannot use the same URL for more than one host.

**Note**

To learn about considerations for setting up a host in a VPC, see **Prerequisites: Network or Amazon VPC configuration for your connection (p. 60).**

**To edit a host**

2. Choose **Settings > Connections**.
3. Choose the **Hosts** tab.
   
   The hosts associated with your AWS account and created in the selected AWS Region are displayed.
4. To edit the host name, enter a new value in **Name**.
5. To edit the host endpoint, enter a new value in **URL**.
6. To edit the host VPC configuration, enter new values in **VPC ID**.
7. Choose **Edit host**.
8. The updated settings are displayed. Choose **Set up Pending host**.

**Delete a host**

You can use the Developer Tools console or the `delete-host` command in the AWS Command Line Interface (AWS CLI) to delete a host.

**Topics**

- Delete a host (console) (p. 82)
- Delete a host (CLI) (p. 82)

**Delete a host (console)**

**To delete a host**

2. Choose the **Hosts** tab. In **Name**, choose the name of the host you want to delete.
3. Choose **Delete**.
4. Enter `delete` in the field to confirm, and then choose **Delete**.
   
   **Important**
   
   This action cannot be undone.

**Delete a host (CLI)**

You can use the AWS Command Line Interface (AWS CLI) to delete a host.

To do this, use the `delete-host` command.

**Important**

Before you can delete a host, you must delete all connections associated with the host. After you run the command, the host is deleted. No confirmation dialog box is displayed.

**To delete a host**

- Open a terminal (Linux, macOS, or Unix) or command prompt (Windows). Use the AWS CLI to run the `delete-host` command, specifying the Amazon Resource Name (ARN) of the host that you want to delete.

```
```
This command returns nothing.

View host details

You can use the Developer Tools console or the `get-host` command in the AWS Command Line Interface (AWS CLI) to view details for a host.

To view host details (console)

2. Choose **Settings > Connections**, and then choose the **Hosts** tab.
3. Choose the button next to the host you want to view, and then choose **View details**.
4. The following information appears for your host:
   - The host name.
   - The provider type for your connection.
   - The endpoint of the infrastructure where your provider is installed.
   - The setup status for your host. A host ready for a connection is in **Available** status. If your host was created but setup was not completed, the host might be in a different status.

   The following statuses are available:
   - **PENDING** - The host has completed creation and is ready to start the setup by registering the provider app on the host.
   - **AVAILABLE** - The host has completed creation and setup and is available for use with connections.
   - **ERROR** - There was an error during host creation or registration.
   - **VPC_CONFIG_VPC_INITIALIZING** - The VPC configuration for the host is being created.
   - **VPC_CONFIG_VPC_FAILED_INITIALIZATION** - The VPC configuration for the host encountered error and failed.
   - **VPC_CONFIG_VPC_AVAILABLE** - The VPC configuration for the host has completed setup and is available.
   - **VPC_CONFIG_VPC_DELETING** - The VPC configuration for the host is being deleted.

5. To delete the host, choose **Delete**.
6. If the host is in **Pending** status, to complete the setup, choose **Set up host**. For more information, see [Set up a pending host](p. 80).

**To view host details (CLI)**

- Open a terminal (Linux, macOS, or Unix) or command prompt (Windows), and use the AWS CLI to run the `get-host` command, specifying the Amazon Resource Name (ARN) of the host that you want to view details for.

```bash
```

This command returns the following output.

```json
{
  "Name": "MyHost",
  "Status": "AVAILABLE",
  "ProviderType": "GitHubEnterpriseServer",
  "ProviderEndpoint": "https://test-instance-1.dev/
}
```

**Logging AWS CodeStar Connections API calls with AWS CloudTrail**

AWS CodeStar Connections is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures all API calls for notifications as events. The calls captured include calls from the Developer Tools console and code calls to the AWS CodeStar Connections API operations.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket, including events for notifications. 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 CodeStar Connections, the IP address from which the request was made, who made the request, when it was made, and other details.

For more information, see the [AWS CloudTrail User Guide](#).

**AWS CodeStar Connections information in CloudTrail**

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AWS CodeStar Connections, 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](#) in the [AWS CloudTrail User Guide](#).

For an ongoing record of events in your AWS account, including events for AWS CodeStar Connections, 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 topics in the [AWS CloudTrail User Guide](#):

- **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

All AWS CodeStar Connections actions are logged by CloudTrail and are documented in the AWS CodeStar Connections API reference. For example, calls to the CreateConnection, DeleteConnection and GetConnection actions 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 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 example shows a CloudTrail log entry that demonstrates the CreateConnection action.

```json
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/Mary_Major",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Mary_Major"
  },
  "eventTime": "2020-04-21T01:09:48Z",
  "eventSource": "codestar-connections.amazonaws.com",
  "eventName": "CreateConnection",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "IP",
  "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.163 Safari/537.36",
  "requestParameters": {
    "providerType": "Bitbucket",
    "connectionName": "my-connection"
  },
  "responseElements": {
    "connectionArn": "arn:aws:codestar-connections:us-west-2:123456789012:connection/7EXAMPLE-5da1-4867-960c-4918175ea3ce"
  },
  "requestID": "ac1fbc15-a84f-4568-9f90-f05f1a57749c",
  "eventID": "7548f5b0-7ecf-430f-84bf-72e364644359",
  "readOnly": false,
  "eventType": "AwsApiCall",
}
```
AWS CodeStar Connections and interface VPC endpoints (AWS PrivateLink)

You can establish a private connection between your VPC and AWS CodeStar Connections by creating an interface VPC endpoint. Interface endpoints are powered by AWS PrivateLink, a technology that enables you to privately access AWS CodeStar Connections 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 AWS CodeStar Connections APIs, because traffic between your VPC and AWS CodeStar Connections 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 CodeStar Connections VPC endpoints

Before you set up an interface VPC endpoint for AWS CodeStar Connections, ensure that you review Interface endpoints in the Amazon VPC User Guide.

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

VPC endpoints are supported in all AWS CodeStar Connections Regions.

VPC endpoint concepts

The following are the key concepts for VPC endpoints:

VPC endpoint

The entry point in your VPC that enables you to connect privately to a service. The following are the different types of VPC endpoints. You create the type of VPC endpoint required by the supported service.

- VPC endpoints for AWS CodeStar Connections actions
- VPC endpoints for AWS CodeStar Connections webhooks

AWS PrivateLink

A technology that provides private connectivity between VPCs and services.

VPC endpoints for AWS CodeStar Connections actions

You can manage VPC endpoints for the AWS CodeStar Connections service.

Creating interface VPC endpoints for AWS CodeStar Connections actions

You can create a VPC endpoint for the AWS CodeStar Connections 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.

To start using connections with your VPC, create an interface VPC endpoint for AWS CodeStar Connections. When you create a VPC endpoint for AWS CodeStar Connections, choose AWS Services, and in Service Name, choose:
• `com.amazonaws.region.codestar-connections.api`: This option creates a VPC endpoint for AWS CodeStar Connections API operations. For example, choose this option if your users use the AWS CLI, the AWS CodeStar Connections API, or the AWS SDKs to interact with AWS CodeStar Connections for operations such as `CreateConnection`, `ListConnections`, and `CreateHost`.

For the **Enable DNS name** option, if you select private DNS for the endpoint, you can make API requests to AWS CodeStar Connections using its default DNS name for the Region, for example, `codestar-connections.us-east-1.amazonaws.com`.

**Important**

Private DNS is enabled by default for endpoints created for AWS services and AWS Marketplace Partner services.

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

### Creating a VPC endpoint policy for AWS CodeStar Connections actions

You can attach an endpoint policy to your VPC endpoint that controls access to AWS CodeStar Connections. 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*.

**Note**

The `com.amazonaws.region.codestar-connections.webhooks` endpoint does not support policies.

**Example: VPC endpoint policy for AWS CodeStar Connections actions**

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

```json
{
   "Statement": [
      {
         "Sid": "GetConnectionOnly",
         "Principal": "*",
         "Action": [ "codestar-connections:GetConnection"
         ],
         "Effect": "Allow",
         "Resource": "*"
      }
   ]
}
```

### VPC endpoints for AWS CodeStar Connections webhooks

AWS CodeStar Connections creates webhook endpoints for you when you create or delete a host with VPC configuration. The endpoint name is `com.amazonaws.region.codestar-connections.webhooks`.

With the VPC endpoint for GitHub webhooks, hosts can send event data via webhooks to your integrated AWS services over the Amazon network.
Important
When you set up your host for GitHub Enterprise Server, AWS CodeStar Connections creates a VPC endpoint for webhooks event data for you. If you created your host before November 24, 2020, and you want to use VPC PrivateLink webhook endpoints, you must first delete your host and then create a new host.

AWS CodeStar Connections manages the lifecycle of these endpoints. To delete the endpoint, you must delete the corresponding host resource.

How webhook endpoints for AWS CodeStar Connections hosts are used

The webhook endpoint is where webhooks from third-party repositories are sent for AWS CodeStar Connections processing. A webhook describes a customer action. When you perform a `git push`, the webhook endpoint receives a webhook from the provider detailing the push. For example, AWS CodeStar Connections can notify CodePipeline to start your pipeline.

For cloud providers, such as Bitbucket, or GitHub Enterprise Server hosts that do not use a VPC, the webhook VPC endpoint does not apply because the providers are sending webhooks to AWS CodeStar Connections where the Amazon network is not used.

Troubleshooting connections

The following information might help you troubleshoot common issues with connections to resources in AWS CodeBuild, AWS CodeDeploy, and AWS CodePipeline.

Topics

- I cannot create connections (p. 88)
- I get a permissions error when I try to create or complete a connection (p. 88)
- I get a permissions error when I try to use a connection (p. 89)
- Connection is not in available state or is no longer pending (p. 89)
- Add GitClone permissions for connections (p. 89)
- Host is not in available state (p. 90)
- Troubleshooting a host with connection errors (p. 90)
- I’m unable to create a connection for my host (p. 90)
- Troubleshooting VPC configuration for your host (p. 91)
- Troubleshooting webhook VPC endpoints (PrivateLink) for GitHub Enterprise Server connections (p. 93)
- Troubleshooting for a host created before November 24, 2020 (p. 94)
- Unable to create the connection for a GitHub repository (p. 94)
- Edit your GitHub Enterprise Server connection app permissions (p. 94)
- I want to increase my limits for connections (p. 95)

I cannot create connections

You might not have permissions to create a connection. For more information, see Permissions and examples for AWS CodeStar Connections (p. 120).

I get a permissions error when I try to create or complete a connection

The following error message might be returned when you try to create or view a connection in the CodePipeline console.
User: *username* is not authorized to perform: *permission* on resource: *connection-ARN*

If this message appears, make sure that you have sufficient permissions.

The permissions to create and view connections in the AWS Command Line Interface (AWS CLI) or the AWS Management Console are only part of the permissions that you need to create and complete connections on the console. The permissions required to simply view, edit, or create a connection and then complete the pending connection should be scoped down for users who only need to perform certain tasks. For more information, see Permissions and examples for AWS CodeStar Connections (p. 120).

**I get a permissions error when I try to use a connection**

One or both of the following error messages might be returned if you try to use a connection in the CodePipeline console, even though you have the permissions to list, get, and create permissions.

You have failed to authenticate your account.

User: *username* is not authorized to perform: *codestar-connections:UseConnection* on resource: *connection-ARN*

If this occurs, make sure that you have sufficient permissions.

Make sure you have the permissions to use a connection, including listing the available repositories in the provider location. For more information, see Permissions and examples for AWS CodeStar Connections (p. 120).

**Connection is not in available state or is no longer pending**

If the console displays a message that a connection is not in an available state, choose **Complete connection**.

If you choose to complete the connection and a message appears that the connection is not in a pending state, you can cancel the request because the connection is already in an available state.

**Add GitClone permissions for connections**

When you use an AWS CodeStar connection in a source action and a CodeBuild action, there are two ways the input artifact can be passed to the build:

- The default: The source action produces a zip file that contains the code that CodeBuild downloads.
- Git clone: The source code can be directly downloaded to the build environment.

The Git clone mode allows you to interact with the source code as a working Git repository. To use this mode, you must grant your CodeBuild environment permissions to use the connection.

To add permissions to your CodeBuild service role policy, you create a customer managed policy that you attach to your CodeBuild service role. The following steps create a policy where the UseConnection permission is specified in the action field, and the connection Amazon Resource Name (ARN) is specified in the Resource field.

**To use the console to add the UseConnection permissions**

1. To find the connection ARN for your pipeline, open your pipeline and choose the (i) icon on your source action. The Configuration pane opens, and the connection ARN appears next to **ConnectionArn**. You add the connection ARN to your CodeBuild service role policy.
2. To find your CodeBuild service role, open the build project used in your pipeline and navigate to the **Build details** tab.
3. In the Environment section, choose the Service role link. This opens the AWS Identity and Access Management (IAM) console, where you can add a new policy that grants access to your connection.

4. In the IAM console, choose Attach policies, and then choose Create policy.

Use the following sample policy template. Add your connection ARN in the Resource field, as shown in this example.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "codestar-connections:UseConnection",
            "Resource": "insert connection ARN here"
        }
    ]
}
```

On the JSON tab, paste your policy.

5. Choose Review policy. Enter a name for the policy (for example, connection-permissions), and then choose Create policy.

6. Return to the service role Attach Permissions page, refresh the policy list, and select the policy you just created. Choose Attach policies.

Host is not in available state

If the console displays a message that a host is not in an Available state, choose Set up host.

The first step for host creation results in the created host now in a Pending state. To move the host to an Available state, you must choose to set up the host in the console. For more information, see Set up a pending host (p. 80).

Note
You cannot use the AWS CLI to set up a Pending host.

Troubleshooting a host with connection errors

Connections and hosts can move into the error state if the underlying GitHub app is deleted or modified. Hosts and connections in the error state cannot be recovered and the host must be recreated.

- Actions such as changing the app pem key, changing the app name (after initial creation) will cause the host and all associated connections to go into the error state.

If the console or CLI returns a host or a connection related to a host with an Error state, you might need to perform the following step:

- Delete and recreate the host resource and then reinstall the host registration app. For more information, see Create a host (p. 78).

I’m unable to create a connection for my host

To create a connection or host, the following conditions are required.

- Your host must be in the AVAILABLE state. For more information, see
• Connections must be created in the same Region as the host.

Troubleshooting VPC configuration for your host

When you create a host resource, you must provide network connection or VPC information for the infrastructure where your GitHub Enterprise Server instance is installed. For troubleshooting your VPC or subnet configuration for your host, use the example VPC information shown here as a reference.

**Note**

Use this section for troubleshooting that is related to your GitHub Enterprise Server host configuration within an Amazon VPC. For troubleshooting that is related to your connection that is configured to use the webhook endpoint for VPC (PrivateLink), see Troubleshooting webhook VPC endpoints (PrivateLink) for GitHub Enterprise Server connections (p. 93).

For this example, you would use the following process to configure the VPC and server where your GitHub Enterprise Server instance will be installed:

1. Create a VPC. For more information, see https://docs.aws.amazon.com/vpc/latest/userguide/working-with-vpcs.html#Create-VPC.
2. Create a subnet in your VPC. For more information, see https://docs.aws.amazon.com/vpc/latest/userguide/working-with-vpcs.html#AddASubnet.
3. Launch an instance into your VPC. For more information, see https://docs.aws.amazon.com/vpc/latest/userguide/working-with-vpcs.html#VPC_Launch_Instance.

The following image shows an EC2 instance launched using the GitHub Enterprise AMI.

When you use a VPC for a GitHub Enterprise Server connection, you must provide the following for your infrastructure when you set up your host:

- **VPC ID**: The VPC for the server where your GitHub Enterprise Server instance is installed or a VPC which has access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.
- **Subnet ID or IDs**: The subnet for the server where your GitHub Enterprise Server instance is installed or a subnet with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.
• **Security group or groups:** The security group for the server where your GitHub Enterprise Server instance is installed or a security group with access to your installed GitHub Enterprise Server instance through VPN or Direct Connect.

• **Endpoint:** Have your server endpoint ready and continue to the next step.

For more information about working with VPCs and subnets, see VPC and Subnet Sizing for IPv4 in the Amazon VPC User Guide.

### Topics
- I'm unable to get a host in pending state (p. 92)
- I'm unable to get a host in available state (p. 92)
- My connection/host was working and has stopped working now (p. 92)
- I'm unable to delete my network interfaces (p. 93)

### I'm unable to get a host in pending state

If your host enters the VPC_CONFIG_FAILED_INITIALIZATION state, this is likely because of an issue with the VPC, subnets, or security groups that you have selected for your host.

- The VPC, subnets, and security groups must all belong to the account creating the host.
- The subnets and security groups must belong to the selected VPC.
- Each provided subnet must be in different Availability Zones.
- The user creating the host must have the following IAM permissions:

```plaintext
ec2:CreateNetworkInterface
ec2:CreateTags
ec2:DescribeDhcpOptions
ec2:DescribeNetworkInterfaces
ec2:DescribeSubnets
ec2:DeleteNetworkInterface
ec2:DescribeVpcs
ec2:CreateVpcEndpoint
ec2:DeleteVpcEndpoints
ec2:DescribeVpcEndpoints
```

### I'm unable to get a host in available state

If you are unable to complete the AWS CodeStar Connections app setup for your host, it may be because of an issue with your VPC configurations or your GitHub Enterprise Server instance.

- If you are not using a public certificate authority, you will need to provide a TLS certificate to your host that is used by your GitHub Enterprise Instance. The TLS Certificate value should be the public key of the certificate.
- You need to be an administrator of the GitHub Enterprise Server instance in order to create GitHub apps.

### My connection/host was working and has stopped working now

If a connection/host was working before and is not working now, it could be due to a configuration change in your VPC or the GitHub app has been modified. Check the following:

- The security group attached to the host resource you created for your connection has now changed or no longer has access to the GitHub Enterprise Server. AWS CodeStar Connections requires a security group which has connectivity to the GitHub Enterprise Server instance.
• DNS Server IP has recently changed. You can verify this by checking the DHCP options attached to the
VPC specified in the host resource you created for your connection. Note that if you’ve recently moved
from AmazonProvidedDNS to custom DNS Server or started using a new custom DNS Server, the host/
connection would stop working. In order to fix this, delete your existing host and re-create it, which
would store the latest DNS settings in our database.
• The network ACLs settings have changed and are no longer allowing HTTP connections to the subnet
where your GitHub Enterprise Server infrastructure is located.
• Any configurations of the AWS CodeStar Connections app on your GitHub Enterprise Server have
changed. Modifications to any of the configurations, such as URLs or app secrets, can break the
connectivity between your installed GitHub Enterprise Server instance and AWS CodeStar Connections.

I’m unable to delete my network interfaces

If you are unable to detect your network interfaces, check the following:

• The Network Interfaces created by AWS CodeStar Connections can only be deleted by deleting the
host. They cannot be deleted manually by the user.
• You must have the following permissions:

```
ec2:DescribeNetworkInterfaces
ec2:DeleteNetworkInterface
```

Troubleshooting webhook VPC endpoints (PrivateLink) for
GitHub Enterprise Server connections

When you create a host with VPC configuration, the webhook VPC endpoint is created for you.

**Note**
Use this section for troubleshooting that is related to your connection that is configured to
use the webhook endpoint for VPC (PrivateLink). For troubleshooting that is related to your
GitHub Enterprise Server host configuration within an Amazon VPC, see Troubleshooting VPC
configuration for your host (p. 91).

When you create a connection to an installed provider type, and you have specified that your server
is configured within a VPC, then AWS CodeStar Connections creates your host, and the VPC endpoint
(PrivateLink) for webhooks is created for you. This enables the host to send event data via webhooks
to your integrated AWS services over the Amazon network. For more information, see AWS CodeStar
Connections and interface VPC endpoints (AWS PrivateLink) (p. 86).

**Topics**

• I’m unable to delete my webhook VPC endpoints (p. 93)

I’m unable to delete my webhook VPC endpoints

AWS Codestar Connections manages the lifecycle of the webhook VPC endpoints for your host. If you
want to delete the endpoint, you must do this by deleting the corresponding host resource.

• The webhook VPC endpoints (PrivateLink) created by AWS CodeStar Connections can only be deleted
by deleting the host. They cannot be deleted manually.
• You must have the following permissions:

```
ec2:DescribeNetworkInterfaces
ec2:DeleteNetworkInterface
```
Troubleshooting for a host created before November 24, 2020

As of November 24, 2020, when AWS CodeStar Connections sets up your host, an additional VPC endpoint (PrivateLink) support is set up for you. For hosts created before this update, use this troubleshooting section.

For more information, see AWS CodeStar Connections and interface VPC endpoints (AWS PrivateLink) (p. 86).

Topics
- I have a host that was created before November 24, 2020 and I want to use VPC endpoints (PrivateLink) for webhooks (p. 94)
- I'm unable to get a host in available state (VPC error) (p. 94)

I have a host that was created before November 24, 2020 and I want to use VPC endpoints (PrivateLink) for webhooks

When you set up your host for GitHub Enterprise Server, the webhook endpoint is created for you. Connections now use VPC PrivateLink webhook endpoints. If you created your host before November 24, 2020, and you want to use VPC PrivateLink webhook endpoints, you must first delete your host and then create a new host.

I'm unable to get a host in available state (VPC error)

If your host was created before November 24, 2020, and you are unable to complete the AWS CodeStar Connections app setup for your host, it may be because of an issue with your VPC configurations or your GitHub Enterprise Server instance.

Your VPC will need a NAT Gateway (or outbound internet access) so that your GitHub Enterprise Server instance can send egress network traffic for GitHub webhooks.

Unable to create the connection for a GitHub repository

Problem:

Because a connection to a GitHub repository uses the AWS Connector for GitHub, you need organization owner permissions or admin permissions to the repository to create the connection.

Possible fixes: For information about permission levels for a GitHub repository, see https://docs.github.com/en/free-pro-team@latest/github/setting-up-and-managing-organizations-and-teams/permission-levels-for-an-organization.

Edit your GitHub Enterprise Server connection app permissions

If you installed the app for GitHub Enterprise Server on or before December 23, 2020, you might need to give the app Read-only access to members of the organization. If you are the GitHub app owner, follow these steps to edit the permissions for the app that was installed when your host was created.

Note

You must complete these steps on your GitHub Enterprise Server instance, and you must be the GitHub app owner.

1. In GitHub Enterprise Server, from the drop-down option on your profile photo, choose Settings.
2. Choose Developer settings, and then choose GitHub Apps.
3. In the list of apps, choose the name of the app for your connection, and then choose Permissions and events in the settings display.
4. **Under Organization permissions**, for **Members**, choose **Read-only** from the **Access** drop-down.

5. **In Add a note to users**, add a description of the reason for the update. Choose **Save changes**.

I want to increase my limits for connections

You can request a limit increase for certain limits in AWS CodeStar Connections. For more information, see **Quotas for connections** (p. 95).

**Quotas for connections**

The following tables list the quotas (also referred to as *limits*) for connections in the Developer Tools console.

Quotas in this table apply per AWS Region and can be increased. To request an increase, use the **Support center console**.

It can take up to two weeks to process requests for a quota increase.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of connections per AWS account</td>
<td>250</td>
</tr>
</tbody>
</table>

Quotas in this table are fixed and cannot be changed.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Default limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum characters in connection names</td>
<td>32 characters</td>
</tr>
<tr>
<td>Maximum number of hosts per AWS account</td>
<td>50</td>
</tr>
</tbody>
</table>
Security for features of the Developer Tools console

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 CodeStar Notifications and AWS CodeStar Connections, 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 CodeStar Notifications and AWS CodeStar Connections. The following topics show you how to configure AWS CodeStar Notifications and AWS CodeStar Connections 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 CodeStar Notifications and AWS CodeStar Connections resources.

For more information about security for the services in the Developer Tools console, see the following:

- CodeBuild Security
- CodeCommit Security
- CodeDeploy Security
- CodePipeline Security

Understanding notification contents and security

Notifications provide information about resources to users who are subscribed to the notification rule targets that you configure. This information can include details about your developer tool resources, including repository contents, build statuses, deployment statuses, and pipeline executions.

For example, you can configure a notification rule for a repository in CodeCommit to include comments on commits or pull requests. If so, the notifications sent in response to that rule might contain the line or lines of code referenced in that comment. Similarly, you can configure a notification rule for a build project in CodeBuild to include successes or failures for build states and phases. Notifications sent in response to that rule will contain that information.

You can configure a notification rule for a pipeline in CodePipeline to include information about manual approvals, and notifications sent in response to that rule might contain the name of the person providing that approval. You can configure a notification rule for an application in CodeDeploy to indicate
deployment success, and notifications sent in response to that rule might contain information about the deployment target.

Notifications can include project-specific information such as build statuses, lines of code that have comments, deployment states, and pipeline approvals. So to help ensure the security of your project, make sure that you regularly review both the targets of notification rules and the list of subscribers of the Amazon SNS topics specified as targets. Additionally, the content of notifications sent in response to events might change as additional features are added to the underlying services. This change can happen without notice to already-existing notification rules. Consider reviewing the contents of notification messages periodically to help ensure that you understand what is being sent, as well as to whom it is being sent.

For more information about the event types available for notification rules, see Notification concepts (p. 4).

You can choose to limit the details included in notifications to only what is included in an event. This is referred to as the Basic detail type. These events contain exactly the same information as is sent to Amazon EventBridge and Amazon CloudWatch Events.

Developer Tools console services, such as CodeCommit, might choose to add information about some or all of their event types in notification messages beyond what is available in an event. This supplemental information could be added at any time to enhance current event types or supplement future event types. You can choose to include any supplemental information about the event, if available, in the notification by choosing the Full detail type. For more information, see Detail types (p. 5).

Data protection in AWS CodeStar Notifications and AWS CodeStar Connections

The AWS shared responsibility model applies to data protection in AWS CodeStar Notifications and AWS CodeStar Connections. 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 user accounts 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 personal 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 confidential or sensitive information, such as your customers' email addresses, into tags or free-form fields such as a Name field. This includes when you
work with AWS CodeStar Notifications and AWS CodeStar Connections or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form 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.

Identity and access management for AWS CodeStar Notifications and AWS CodeStar Connections

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 CodeStar Notifications and AWS CodeStar Connections resources. IAM is an AWS service that you can use with no additional charge.

Topics
- Audience (p. 98)
- Authenticating with identities (p. 99)
- Managing access using policies (p. 101)
- How features in the developer tools console work with IAM (p. 101)
- AWS CodeStar Connections permissions reference (p. 105)
- Identity-based policy examples (p. 115)
- Using tags to control access to AWS CodeStar Connections resources (p. 124)
- Policy best practices (p. 125)
- Using notifications and connections in the console (p. 125)
- Allow users to view their own permissions (p. 126)
- Troubleshooting AWS CodeStar Notifications and AWS CodeStar Connections identity and access (p. 127)
- Using service-linked roles for AWS CodeStar Notifications (p. 129)

Audience

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

Service user – If you use the AWS CodeStar Notifications and AWS CodeStar Connections service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AWS CodeStar Notifications and AWS CodeStar Connections 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 CodeStar Notifications and AWS CodeStar Connections, see Troubleshooting AWS CodeStar Notifications and AWS CodeStar Connections identity and access (p. 127).

Service administrator – If you're in charge of AWS CodeStar Notifications and AWS CodeStar Connections resources at your company, you probably have full access to AWS CodeStar Notifications and AWS CodeStar Connections. It's your job to determine which AWS CodeStar Notifications and AWS CodeStar Connections features and resources your employees 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 CodeStar Notifications and AWS CodeStar Connections, see How features in the developer tools console work with IAM (p. 101).

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 CodeStar Notifications and AWS CodeStar Connections. To view example AWS CodeStar Notifications and AWS CodeStar Connections identity-based policies that you can use in IAM, see Identity-based policy examples (p. 115).

## Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see Signing in to the AWS Management Console as an IAM user or root user in the IAM User Guide.

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 also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the AWS Management Console, use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request using your credentials. If you don’t use AWS tools, you must sign the request yourself. Do this using Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, 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 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 do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

### IAM users and groups

An IAM user is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see Managing access keys for IAM users in the IAM User Guide. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

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:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.

- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as federated users. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated users and roles in the IAM User Guide.

- **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. To see whether an action requires additional dependent actions in a policy, see Actions, Resources, and Condition Keys for AWS CodeStar Notifications and Actions, Resources, and Condition Keys for AWS CodeStar Connections in the Service Authorization Reference.

- **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 IAM 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 EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an 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 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 or IAM users, 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 IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. 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. In other words, 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, 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 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.

How features in the developer tools console work with IAM

Before you use IAM to manage access to features in the Developer Tools console, you should understand which IAM features are available to use with it. To get a high-level view of how notifications and other AWS services work with IAM, see AWS services that work with IAM in the IAM User Guide.

Topics

- Identity-based policies in the developer tools console (p. 101)
- AWS CodeStar Notifications and AWS CodeStar Connections resource-based policies (p. 104)
- Authorization based on tags (p. 104)
- IAM roles (p. 105)

Identity-based policies in the developer tools console

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. AWS CodeStar Notifications and AWS CodeStar
Connections support specific actions, resources, and condition keys. 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

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. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as permission-only actions that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called dependent actions.

Include actions in a policy to grant permissions to perform the associated operation.

Policy actions for notifications in the Developer Tools console use the following prefixes before the action: codestar-notifications and codestar-connections. For example, to grant someone permission to view all notification rules in their account, you include the codestar-notifications:ListNotificationRules action in their policy. Policy statements must include either an Action or NotAction element. AWS CodeStar Notifications and AWS CodeStar Connections defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple AWS CodeStar Notifications actions in a single statement, separate them with commas as follows.

```
"Action": [ "codestar-notifications:action1", "codestar-notifications:action2"
```

To specify multiple AWS CodeStar Connections actions in a single statement, separate them with commas as follows.

```
"Action": [ "codestar-connections:action1", "codestar-connections:action2"
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word List, include the following action.

```
"Action": "codestar-notifications:List*"
```

AWS CodeStar Notifications API actions include:

- CreateNotificationRule
- DeleteNotificationRule
- DeleteTarget
- DescribeNotificationRule
- ListEventTypes
- ListNotificationRules
- ListTagsForResource
- ListTargets
- Subscribe
• TagResource
• Unsubscribe
• UntagResource
• UpdateNotificationRule

AWS CodeStar Connections API actions include the following:

• CreateConnection
• DeleteConnection
• GetConnection
• ListConnections
• ListTagsForResource
• TagResource
• UntagResource

The following permissions-only actions are required in AWS CodeStar Connections to complete the auth handshake:

• GetIndividualAccessToken
• GetInstallationUrl
• ListInstallationTargets
• StartOAuthHandshake
• UpdateConnectionInstallation

The following permissions-only action is required in AWS CodeStar Connections to use a connection:

• UseConnection

The following permissions-only action is required in AWS CodeStar Connections to pass a connection to a service:

• PassConnection

To see a list of AWS CodeStar Notifications and AWS CodeStar Connections actions, see Actions Defined by AWS CodeStar Notifications and Actions Defined by AWS CodeStar Connections in the IAM User Guide.

Resources

AWS CodeStar Notifications and AWS CodeStar Connections do not support specifying resource ARNs in a policy.

Condition keys

AWS CodeStar Notifications and AWS CodeStar Connections define their own sets of condition keys and also support using some global condition keys. To see all AWS global condition keys, see AWS global condition context keys in the IAM User Guide.

All AWS CodeStar Notifications actions support the codestar-notifications:NotificationsForResource condition key. For more information, see Identity-based policy examples (p. 115).
AWS CodeStar Connections define the following condition keys that can be used in the Condition element of an IAM policy. You can use these keys to further refine the conditions under which the policy statement applies. For more information, see AWS CodeStar Connections permissions reference (p. 105).

<table>
<thead>
<tr>
<th>Condition keys</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:BranchName</td>
<td>Filters access by the third-party repository branch name</td>
</tr>
<tr>
<td>codestar-connections:FullRepositoryId</td>
<td>Filters access by the repository that is passed in the request. Applies only to UseConnection requests for access to a specific repository</td>
</tr>
<tr>
<td>codestar-connections:InstallationId</td>
<td>Filters access by the third-party ID (such as the Bitbucket app installation ID) that is used to update a connection. Allows you to restrict which third-party app installations can be used to make a connection</td>
</tr>
<tr>
<td>codestar-connections:OwnerId</td>
<td>Filters access by the owner or account ID of the third-party provider</td>
</tr>
<tr>
<td>codestar-connections:PassedToService</td>
<td>Filters access by the service to which the principal is allowed to pass a connection</td>
</tr>
<tr>
<td>codestar-connections:ProviderAction</td>
<td>Filters access by the provider action in a UseConnection request such as ListRepositories.</td>
</tr>
<tr>
<td>codestar-connections:ProviderPermissionsRequired</td>
<td>Filters access by the type of third-party provider permissions</td>
</tr>
<tr>
<td>codestar-connections:ProviderType</td>
<td>Filters access by the type of third-party provider passed in the request</td>
</tr>
<tr>
<td>codestar-connections:ProviderTypeFilter</td>
<td>Filters access by the type of third-party provider used to filter results</td>
</tr>
<tr>
<td>codestar-connections:RepositoryName</td>
<td>Filters access by the third-party repository name</td>
</tr>
</tbody>
</table>

**Examples**

To view examples of AWS CodeStar Notifications and AWS CodeStar Connections identity-based policies, see Identity-based policy examples (p. 115).

**AWS CodeStar Notifications and AWS CodeStar Connections resource-based policies**

AWS CodeStar Notifications and AWS CodeStar Connections do not support resource-based policies.

**Authorization based on tags**

You can attach tags to AWS CodeStar Notifications and AWS CodeStar Connections resources or pass tags in a request. To control access based on tags, you provide tag information in the condition element of a policy using the codestar-notifications and codestar-
connections:ResourceTag/key-name, aws:RequestTag/key-name, or aws:TagKeys condition keys. For more information about tagging strategies, see Tagging AWS resources. For more information about tagging AWS CodeStar Notifications and AWS CodeStar Connections resources, see Tag connections resources (p. 69).

To view example identity-based policies for limiting access to a resource based on the tags on that resource, see Using tags to control access to AWS CodeStar Connections resources (p. 124).

IAM roles

An IAM role is an entity within your AWS account that has specific permissions.

Using temporary credentials

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

AWS CodeStar Notifications and AWS CodeStar Connections supports the use of temporary credentials.

Service-linked roles

Service-linked roles allow AWS services to access resources in other services to complete an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view but not edit the permissions for service-linked roles.

AWS CodeStar Notifications supports service-linked roles. For details about creating or managing AWS CodeStar Notifications and AWS CodeStar Connections service-linked roles, see Using service-linked roles for AWS CodeStar Notifications (p. 129).

AWS CodeStar Connections does not support service-linked roles.

**AWS CodeStar Connections permissions reference**

The following tables list each AWS CodeStar Connections API operation, the corresponding actions for which you can grant permissions, and the format of the resource ARN to use for granting permissions. The AWS CodeStar Connections APIs are grouped into tables based on the scope of the actions allowed by that API. Refer to it when writing permissions policies that you can attach to an IAM identity (identity-based policies).

When you create a permissions policy, you specify the actions in the policy's Action field. You specify the resource value in the policy's Resource field as an ARN, with or without a wildcard character (*).

To express conditions in your connections policies, use the condition keys described here and listed in Condition keys (p. 103). You can also use AWS-wide condition keys. For a complete list of AWS-wide keys, see Available keys in the IAM User Guide.

To specify an action, use the codestar-connections: prefix followed by the API operation name (for example, codestar-connections:ListConnections or codestar-connections:CreateConnection.

**Using wildcards**

To specify multiple actions or resources, use a wildcard character (*) in your ARN. For example, codestar-connections:* specifies all AWS CodeStar Connections actions and codestar-connections:Get* specifies all AWS CodeStar Connections actions that begin with the word Get. The following example grants access to all resources with names that begin with MyConnection.

```
```
You can use wildcards only with the `connection` resources listed in the following table. You can't use wildcards with `region` or `account-id` resources. For more information about wildcards, see IAM identifiers in IAM User Guide.

**Topics**
- Permissions for managing connections (p. 106)
- Permissions for managing hosts (p. 107)
- Permissions for completing connections (p. 108)
- Permissions for setting up hosts (p. 110)
- Passing a connection to a service (p. 111)
- Using a connection (p. 111)
- Supported access types for ProviderAction (p. 112)
- Supported permissions for tagging connection resources (p. 114)

### Permissions for managing connections

A role or user designated to use the AWS CLI or SDK to view, create, or delete connections should have permissions limited to the following.

**Note**
You cannot complete or use a connection in the console with only the following permissions. You need to add the permissions in Permissions for completing connections (p. 108).

<table>
<thead>
<tr>
<th>Action(s)</th>
<th>Required to use the CLI or console to create a connection.</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:CreateConnection</td>
<td></td>
</tr>
<tr>
<td>codestar-connections:DeleteConnection</td>
<td></td>
</tr>
<tr>
<td>codestar-connections:GetConnection</td>
<td></td>
</tr>
<tr>
<td>codestar-connections:ListConnections</td>
<td></td>
</tr>
</tbody>
</table>

### AWS CodeStar Notifications and AWS CodeStar Connections required permissions for actions for managing connections

**CreateConnection**

**Action(s):** codestar-connections:CreateConnection

Required to use the CLI or console to create a connection.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

**DeleteConnection**

**Action(s):** codestar-connections:DeleteConnection

Required to use the CLI or console to delete a connection.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

**GetConnection**

**Action(s):** codestar-connections:GetConnection

Required to use the CLI or console to view details about a connection.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id
ListConnections

**Action(s):** codestar-connections:ListConnections

Required to use the CLI or console to list all connections in the account.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

These operations support the following condition keys:

<table>
<thead>
<tr>
<th>Action</th>
<th>Condition keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:CreateConnection</td>
<td>codestar-connections:ProviderType</td>
</tr>
<tr>
<td>codestar-connections:DeleteConnection</td>
<td>N/A</td>
</tr>
<tr>
<td>codestar-connections:GetConnection</td>
<td>N/A</td>
</tr>
<tr>
<td>codestar-connections:ListConnections</td>
<td>codestar-connections:ProviderTypeFilter</td>
</tr>
</tbody>
</table>

**Permissions for managing hosts**

A role or user designated to use the AWS CLI or SDK to view, create, or delete hosts should have permissions limited to the following.

**Note**

You cannot complete or use a connection in the host with only the following permissions. You need to add the permissions in Permissions for setting up hosts (p. 110).

<table>
<thead>
<tr>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:CreateHost</td>
</tr>
<tr>
<td>codestar-connections:DeleteHost</td>
</tr>
<tr>
<td>codestar-connections:GetHost</td>
</tr>
<tr>
<td>codestar-connections:ListHosts</td>
</tr>
</tbody>
</table>

**AWS CodeStar Notifications and AWS CodeStar Connections required permissions for actions for managing hosts**

CreateHost

**Action(s):** codestar-connections:CreateHost

Required to use the CLI or console to create a host.

**Resource:** arn:aws:codestar-connections:region:account-id:host/host-id

DeleteHost

**Action(s):** codestar-connections:DeleteHost

Required to use the CLI or console to delete a host.

**Resource:** arn:aws:codestar-connections:region:account-id:host/host-id

GetHost

**Action(s):** codestar-connections:GetHost

Required to use the CLI or console to view details about a host.
**Resource:** `arn:aws:codestar-connections:region:account-id:host/host-id`

**ListHosts**

**Action(s):** `codestar-connections:ListHosts`

Required to use the CLI or console to list all hosts in the account.

**Resource:** `arn:aws:codestar-connections:region:account-id:host/host-id`

These operations support the following condition keys:

<table>
<thead>
<tr>
<th>Action</th>
<th>Condition keys</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>codestar-connections:CreateHost</code></td>
<td><code>codestar-connections:ProviderType</code></td>
</tr>
<tr>
<td><code>codestar-connections:DeleteHost</code></td>
<td>N/A</td>
</tr>
<tr>
<td><code>codestar-connections:GetHost</code></td>
<td>N/A</td>
</tr>
<tr>
<td><code>codestar-connections:ListHosts</code></td>
<td><code>codestar-connections:ProviderTypeFilter</code></td>
</tr>
</tbody>
</table>

**Permissions for completing connections**

A role or user designated to manage connections in the console should have the permissions required to complete a connection in the console and create an installation, which includes authorizing the handshake to the provider and creating installations for connections to use. Use the following permissions in addition to the permissions above.

The following IAM operations are used by the console when performing a browser-based handshake. The `ListInstallationTargets`, `GetInstallationUrl`, `StartOAuthHandshake`, `UpdateConnectionInstallation`, and `GetIndividualAccessToken` are IAM policy permissions. They are not API actions.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>codestar-connections:GetInstallationUrl</code></td>
<td><code>codestar-connections:GetInstallationUrl</code></td>
</tr>
<tr>
<td><code>codestar-connections:ListInstallationTargets</code></td>
<td><code>codestar-connections:ListInstallationTargets</code></td>
</tr>
<tr>
<td><code>codestar-connections:StartOAuthHandshake</code></td>
<td><code>codestar-connections:StartOAuthHandshake</code></td>
</tr>
<tr>
<td><code>codestar-connections:UpdateConnectionInstallation</code></td>
<td><code>codestar-connections:UpdateConnectionInstallation</code></td>
</tr>
</tbody>
</table>

Based on this, the following permissions are needed to use, create, update, or delete a connection in the console.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Permission</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>codestar-connections:CreateConnection</code></td>
<td><code>codestar-connections:CreateConnection</code></td>
</tr>
<tr>
<td><code>codestar-connections:DeleteConnection</code></td>
<td><code>codestar-connections:DeleteConnection</code></td>
</tr>
<tr>
<td><code>codestar-connections:GetConnection</code></td>
<td><code>codestar-connections:GetConnection</code></td>
</tr>
<tr>
<td><code>codestar-connections:ListConnections</code></td>
<td><code>codestar-connections:ListConnections</code></td>
</tr>
<tr>
<td><code>codestar-connections:UseConnection</code></td>
<td><code>codestar-connections:UseConnection</code></td>
</tr>
<tr>
<td><code>codestar-connections:ListInstallationTargets</code></td>
<td><code>codestar-connections:ListInstallationTargets</code></td>
</tr>
<tr>
<td><code>codestar-connections:GetInstallationUrl</code></td>
<td><code>codestar-connections:GetInstallationUrl</code></td>
</tr>
<tr>
<td><code>codestar-connections:StartOAuthHandshake</code></td>
<td><code>codestar-connections:StartOAuthHandshake</code></td>
</tr>
<tr>
<td><code>codestar-connections:UpdateConnectionInstallation</code></td>
<td><code>codestar-connections:UpdateConnectionInstallation</code></td>
</tr>
</tbody>
</table>
GetIndividualAccessToken

**Action(s):** codestar-connections:GetIndividualAccessToken

Required to use the console to complete a connection. This is an IAM policy permission only, not an API action.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

GetInstallationUrl

**Action(s):** codestar-connections:GetInstallationUrl

Required to use the console to complete a connection. This is an IAM policy permission only, not an API action.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

ListInstallationTargets

**Action(s):** codestar-connections:ListInstallationTargets

Required to use the console to complete a connection. This is an IAM policy permission only, not an API action.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

StartOAuthHandshake

**Action(s):** codestar-connections:StartOAuthHandshake

Required to use the console to complete a connection. This is an IAM policy permission only, not an API action.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

UpdateConnectionInstallation

**Action(s):** codestar-connections:UpdateConnectionInstallation

Required to use the console to complete a connection. This is an IAM policy permission only, not an API action.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

These operations support the following condition keys.

<table>
<thead>
<tr>
<th>Action</th>
<th>Condition keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:GetIndividualAccessToken</td>
<td>codestar-connections:ProviderType</td>
</tr>
<tr>
<td>codestar-connections:GetInstallationUrl</td>
<td>codestar-connections:ProviderType</td>
</tr>
<tr>
<td>codestar-connections:ListInstallationTargets</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Permissions for setting up hosts

A role or user designated to manage connections in the console should have the permissions required to set up a host in the console, which includes authorizing the handshake to the provider and installing the host app. Use the following permissions in addition to the permissions for hosts above.

The following IAM operations are used by the console when performing a browser-based host registration. RegisterAppCode and StartAppRegistrationHandshake are IAM policy permissions. They are not API actions.

Based on this, the following permissions are needed to use, create, update, or delete a connection in the console that requires a host (such as installed provider types).

**AWS CodeStar Connections required permissions for actions for completing host setup**

**RegisterAppCode**

- **Action(s):** codestar-connections:RegisterAppCode
  - Required to use the console to complete host setup. This is an IAM policy permission only, not an API action.
  - **Resource:** arn:aws:codestar-connections:region:account-id:host/host-id

**StartAppRegistrationHandshake**

- **Action(s):** codestar-connections:StartAppRegistrationHandshake
  - Required to use the console to complete host setup. This is an IAM policy permission only, not an API action.
  - **Resource:** arn:aws:codestar-connections:region:account-id:host/host-id

These operations support the following condition keys.
Passing a connection to a service

When a connection is passed to a service (for example, when a connection ARN is provided in a pipeline definition to create or update a pipeline) the user must have the `codestar-connections:PassConnection` permission.

**AWS CodeStar Connections required permissions for passing a connection**

**PassConnection**

- **Action(s):** codestar-connections:PassConnection
- Required to pass a connection to a service.

This operation also supports the following condition key:

- codestar-connections:PassedToService

### Supported values for condition keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Valid action providers</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:PassedToService</td>
<td>• codeguru-reviewer</td>
</tr>
<tr>
<td></td>
<td>• codepipeline.amazonaws.com</td>
</tr>
<tr>
<td></td>
<td>• proton.amazonaws.com</td>
</tr>
</tbody>
</table>

Using a connection

When a service like CodePipeline uses a connection, the service role must have the `codestar-connections:UseConnection` permission for a given connection.

To manage connections in the console, the user policy must have the `codestar-connections:UseConnection` permission.

**AWS CodeStar Connections required action for using a connection**

**UseConnection**

- **Action(s):** codestar-connections:UseConnection
- Required to use a connection.

This operation also supports the following condition keys:

- codestar-connections:BranchName
- codestar-connections:FullRepositoryId
- codestar-connections:OwnerId
- codestar-connections:ProviderAction
- codestar-connections:ProviderPermissionsRequired
• codestar-connections:RepositoryName

**Supported values for condition keys**

<table>
<thead>
<tr>
<th>Key</th>
<th>Valid action providers</th>
</tr>
</thead>
<tbody>
<tr>
<td>codestar-connections:FullRepositoryId</td>
<td>The user name and repository name of a Bitbucket repository, such as my-owner/my-repository. Supported only when the connection is being used to access a specific repository.</td>
</tr>
<tr>
<td>codestar-connections:ProviderPermissionsRequired</td>
<td>read_only or read_write</td>
</tr>
</tbody>
</table>

The required condition keys for some functionality might change over time. We recommend that you use codestar-connections:UseConnection to control access to a connection unless your access control requirements require different permissions.

**Supported access types for ProviderAction**

When a connection is used by an AWS service, it results in API calls being made to your source code provider. For example, a service might list repositories for a Bitbucket connection by calling the https://api.bitbucket.org/2.0/repositories/username API.

The ProviderAction condition key allows you to restrict which APIs on a provider can be called. Because the API path might be generated dynamically, and the path varies from provider to provider, the ProviderAction value is mapped to an abstract action name rather than the URL of the API. This allows you to write policies that have the same effect regardless of the provider type for the connection.

The following are the access types that are granted for each of the supported ProviderAction values. The following are IAM policy permissions. They are not API actions.

**AWS CodeStar Connections supported access types for ProviderAction**

*GetBranch*

**Action(s):** codestar-connections:GetBranch

Required to access information about a branch, such as the latest commit for that branch.

**Resource:** arn:aws:codestar-connections:region:account-id:connection/connection-id

*ListRepositories*

**Action(s):** codestar-connections:ListRepositories
Required to access a list of public and private repositories, including details about those repositories, that belong to an owner.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListOwners**

**Action(s):** `codestar-connections:ListOwners`

Required to access a list of owners that the connection has access to.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListBranches**

**Action(s):** `codestar-connections:ListBranches`

Required to access the list of branches that exist on a given repository.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**StartUploadArchiveToS3**

**Action(s):** `codestar-connections:StartUploadArchiveToS3`

Required to read source code and upload it to Amazon S3.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**GitPush**

**Action(s):** `codestar-connections:GitPush`

Required to write to a repository using Git.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**GitPull**

**Action(s):** `codestar-connections:GitPull`

Required to read from a repository using Git.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**GetUploadArchiveToS3Status**

**Action(s):** `codestar-connections:GetUploadArchiveToS3Status`

Required to access the status of an upload, including any error messages, started by `StartUploadArchiveToS3`.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**CreatePullRequestDiffComment**

**Action(s):** `codestar-connections:CreatePullRequestDiffComment`

Required to access comments on a pull request.
**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**GetPullRequest**

**Action(s):** `codestar-connections:GetPullRequest`

Required to view pull requests for a repository.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListBranchCommits**

**Action(s):** `codestar-connections:ListBranchCommits`

Required to view a list of commits for a repository branch.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListCommitFiles**

**Action(s):** `codestar-connections:ListCommitFiles`

Required to view a list of files for a commit.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListPullRequestComments**

**Action(s):** `codestar-connections:ListPullRequestComments`

Required to view a list of comments for a pull request.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**ListPullRequestCommits**

**Action(s):** `codestar-connections:ListPullRequestCommits`

Required to view a list of commits for a pull request.

**Resource:** `arn:aws:codestar-connections:region:account-id:connection/connection-id`

**Supported permissions for tagging connection resources**

The following IAM operations are used when tagging connection resources.

- `codestar-connections:ListTagsForResource`
- `codestar-connections:TagResource`
- `codestar-connections:UntagResource`

**AWS CodeStar Connections required actions for tagging connection resources**

**ListTagsForResource**

**Action(s):** `codestar-connections:ListTagsForResource`

Required to view a list of tags associated with the connection resource.
Identity-based policy examples

By default, IAM users and roles who have one of the managed policies for AWS CodeCommit, AWS CodeBuild, AWS CodeDeploy, or AWS CodePipeline applied have permissions to connections, notifications, and notification rules that align with the intent of those policies. For example, IAM users or roles that have one of the full access policies (AWSCodeCommitFullAccess, AWSCodeBuildAdminAccess, AWSCodeDeployFullAccess, or AWSCodePipeline_FullAccess) applied to them also have full access to notifications and notification rules created for the resources for those services.

Other IAM users and roles don't have permission to create or modify AWS CodeStar Notifications and AWS CodeStar Connections 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 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.

Permissions and examples for AWS CodeStar Notifications

The following policy statements and examples can help you manage AWS CodeStar Notifications.

Permissions related to notifications in full access managed policies

The AWSCodeCommitFullAccess, AWSCodeBuildAdminAccess, AWSCodeDeployFullAccess, and AWSCodePipeline_FullAccess managed policies include the following statements to allow full access to notifications in the Developer Tools console. Users with one of these managed policies applied can also create and manage Amazon SNS topics for notifications, subscribe and unsubscribe users to topics, and list topics to choose as targets for notification rules.

Note
In the managed policy, the condition key codestar-notifications:NotificationsForResource will have a value specific to the resource type for the service. For example, in the full access policy for CodeCommit, the value is arn:aws:codecommit:*.
"Sid": "CodeStarNotificationsReadWriteAccess",
"Effect": "Allow",
"Action": [
  "codestar-notifications:CreateNotificationRule",
  "codestar-notifications:DescribeNotificationRule",
  "codestar-notifications:UpdateNotificationRule",
  "codestar-notifications:DeleteNotificationRule",
  "codestar-notifications:Subscribe",
  "codestar-notifications:Unsubscribe"
],
"Resource": "*",
"Condition": {
  "StringLike": {"codestar-notifications:NotificationsForResource": "arn:aws:<vendor-code>:***"}
}
},
{"Sid": "CodeStarNotificationsListAccess",
"Effect": "Allow",
"Action": [
  "codestar-notifications:ListNotificationRules",
  "codestar-notifications:ListTargets",
  "codestar-notifications:ListTagsforResource",
  "codestar-notifications:ListEventTypes"
],
"Resource": "***"}
},
{"Sid": "CodeStarNotificationsSNSTopicCreateAccess",
"Effect": "Allow",
"Action": [
  "sns:CreateTopic",
  "sns:SetTopicAttributes"
],
"Resource": "arn:aws:sns:***:codestar-notifications***"
},
{"Sid": "SNSTopicListAccess",
"Effect": "Allow",
"Action": [
  "sns:ListTopics"
],
"Resource": "***"}
},
{"Sid": "CodeStarNotificationsChatbotAccess",
"Effect": "Allow",
"Action": [
  "chatbot:DescribeSlackChannelConfigurations"
],
"Resource": "***"}

Permissions related to notifications in read-only managed policies

The AWSCodeCommitReadOnlyAccess, AWSCodeBuildReadOnlyAccess, AWSCodeDeployReadOnlyAccess, and AWSCodePipeline_ReadOnlyAccess managed policies include the following statements to allow read-only access to notifications. For example, they can view notifications for resources in the Developer Tools console, but cannot create, manage, or subscribe to them.

Note
In the managed policy, the condition key codestar-notifications:NotificationsForResource will have a value specific to the resource.
type for the service. For example, in the full access policy for CodeCommit, the value is `arn:aws:codecommit:*`.

```json
{
  "Sid": "CodeStarNotificationsPowerUserAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:DescribeNotificationRule"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {"codestar-notifications:NotificationsForResource": 
  "arn:aws:<vendor-code>:***"
  }
}
{
  "Sid": "CodeStarNotificationsListAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:ListNotificationRules",
    "codestar-notifications:ListEventTypes",
    "codestar-notifications:ListTargets"
  ],
  "Resource": "*"
}
```

Permissions related to notifications in other managed policies

The `AWSCodeCommitPowerUser`, `AWSCodeBuildDeveloperAccess`, and `AWSCodeBuildDeveloperAccess` managed policies include the following statements to allow developers with one of these managed policies applied to create, edit, and subscribe to notifications. They cannot delete notification rules or manage tags for resources.

**Note**

In the managed policy, the condition key `codestar-notifications:NotificationsForResource` will have a value specific to the resource type for the service. For example, in the full access policy for CodeCommit, the value is `arn:aws:codecommit:*`.

```json
{
  "Sid": "CodeStarNotificationsReadWriteAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:CreateNotificationRule",
    "codestar-notifications:DescribeNotificationRule",
    "codestar-notifications:UpdateNotificationRule",
    "codestar-notifications:Subscribe",
    "codestar-notifications:Unsubscribe"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {"codestar-notifications:NotificationsForResource": 
  "arn:aws:<vendor-code>:***"
  }
}
{
  "Sid": "CodeStarNotificationsListAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:ListNotificationRules",
    "codestar-notifications:ListTargets",
    "codestar-notifications:ListTagsForResource",
    "codestar-notifications:ListEventTypes"
  ]
}
Example: An administrator-level policy for managing AWS CodeStar Notifications

In this example, you want to grant an IAM user in your AWS account full access to AWS CodeStar Notifications so that the user can review details of notification rules and list notification rules, targets, and event types. You also want to allow the user to add, update, and delete notification rules. This is a full access policy, equivalent to the notification permissions included as part of the `AWSCodeBuildAdminAccess`, `AWSCodeCommitFullAccess`, `AWSCodeDeployFullAccess`, and `AWSCodePipeline_FullAccess` managed policies. Like those managed policies, you should only attach this kind of policy statement to IAM users, groups, or roles that require full administrative access to notifications and notification rules across your AWS account.

**Note**

This policy contains allows `CreateNotificationRule`. Any user with this policy applied to their IAM user or role will be able to create notification rules for any and all resource types supported by AWS CodeStar Notifications in the AWS account, even if that user does not have access to those resources themselves. For example, a user with this policy could create a notification rule for a CodeCommit repository without having permissions to access CodeCommit itself.

```json
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "AWSCodeStarNotificationsFullAccess",
         "Effect": "Allow",
         "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe",
            "codestar-notifications:DeleteTarget",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsForResource",
            "codestar-notifications:TagResource",
            "codestar-notifications:UntagResource"
         ],
         "Resource": "*
      }
```
Example: A contributor-level policy for using AWS CodeStar Notifications

In this example, you want to grant access to the day-to-day usage of AWS CodeStar Notifications, such as creating and subscribing to notifications, but not to more destructive actions, such as deleting notification rules or targets. This is the equivalent to the access provided in the AWSCodeBuildDeveloperAccess, AWSCodeDeployDeveloperAccess, and AWSCodeCommitPowerUser managed policies.

**Note**

This policy contains allows CreateNotificationRule. Any user with this policy applied to their IAM user or role will be able to create notification rules for any and all resource types supported by AWS CodeStar Notifications in the AWS account, even if that user does not have access to those resources themselves. For example, a user with this policy could create a notification rule for a CodeCommit repository without having permissions to access CodeCommit itself.

```json
{
  "Version": "2012-10-17",
  "Sid": "AWSCodeStarNotificationsPowerUserAccess",
  "Effect": "Allow",
  "Action": [
    "codestar-notifications:CreateNotificationRule",
    "codestar-notifications:DescribeNotificationRule",
    "codestar-notifications:ListNotificationRules",
    "codestar-notifications:UpdateNotificationRule",
    "codestar-notifications:Subscribe",
    "codestar-notifications:Unsubscribe",
    "codestar-notifications:ListTargets",
    "codestar-notifications:ListTagsforResource"
  ],
  "Resource": "*"
}
```

Example: A read-only-level policy for using AWS CodeStar Notifications

In this example, you want to grant an IAM user in your account read-only access to the notification rules, targets, and event types in your AWS account. This example shows how you might create a policy that allows viewing these items. This is the equivalent to the permissions included as part of the AWSCodeBuildReadOnlyAccess, AWSCodeCommitReadOnly, and AWSCodePipeline_ReadOnlyAccess managed policies.

```json
{
  "Version": "2012-10-17",
  "Id": "CodeNotification__ReadOnly",
  "Statement": [
    {
      "Sid": "Reads_API_Access",
      "Effect": "Allow",
      "Action": [
        "CodeNotification:DescribeNotificationRule",
        "CodeNotification:ListNotificationRules",
        "CodeNotification:ListTargets",
        "CodeNotification:ListEventTypes"
      ],
      "Resource": "*"
    }
  ]
}
```
Permissions and examples for AWS CodeStar Connections

The following policy statements and examples can help you manage AWS CodeStar Connections.

For information about 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.

Example: A policy for creating AWS CodeStar Connections with the CLI and viewing with the console

A role or user designated to use the AWS CLI or SDK to view, create, tag, or delete connections should have permissions limited to the following.

**Note**
You cannot complete a connection in the console with only the following permissions. You need to add the permissions in the next section.

To use the console to view a list of available connections, view tags, and use a connection, use the following policy.

```
{  
  "Version": "2012-10-17",  
  "Statement": [   
    {   
      "Sid": "ConnectionsFullAccess",  
      "Effect": "Allow",  
      "Action": [   
        "codestar-connections:CreateConnection",  
        "codestar-connections:DeleteConnection",  
        "codestar-connections:UseConnection",  
        "codestar-connections:GetConnection",  
        "codestar-connections:ListConnections",  
        "codestar-connections:TagResource",  
        "codestar-connections:ListTagsForResource",  
        "codestar-connections:UntagResource"
      ],
      "Resource": "*"
    }
  ]
}
```

Example: A policy for creating AWS CodeStar Connections with the console

A role or user designated to manage connections in the console should have the permissions required to complete a connection in the console and create an installation, which includes authorizing the handshake to the provider and creating installations for connections to use. UseConnection should also be added to use the connection in the console. Use the following policy to view, use, create, tag, or delete a connection in the console.

```
{  
  "Version": "2012-10-17",  
  "Statement": [   
    {   
      "Effect": "Allow",  
      "Action": [   
        "codestar-connections:CreateConnection",  
        "codestar-connections:DeleteConnection",
        "codestar-connections:UseConnection",

```
Example: An administrator-level policy for managing AWS CodeStar Connections

In this example, you want to grant an IAM user in your AWS account full access to AWS CodeStar connections so that the user can add, update, and delete connections. This is a full access policy, equivalent to the AWSCodePipeline_FullAccess managed policy. Like that managed policy, you should only attach this kind of policy statement to IAM users, groups, or roles that require full administrative access to connections across your AWS account.

```json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ConnectionsFullAccess",
            "Effect": "Allow",
            "Action": [
                "codestar-connections:CreateConnection",
                "codestar-connections:DeleteConnection",
                "codestar-connections:UseConnection",
                "codestar-connections:GetConnection",
                "codestar-connections:ListConnections",
                "codestar-connections:ListInstallationTargets",
                "codestar-connections:GetInstallationUrl",
                "codestar-connections:StartOAuthHandshake",
                "codestar-connections:UpdateConnectionInstallation",
                "codestar-connections:GetIndividualAccessToken",
                "codestar-connections:TagResource",
                "codestar-connections:ListTagsForResource",
                "codestar-connections:UntagResource"
            ],
            "Resource": "*"
        }
    ]
}
```

Example: A contributor-level policy for using AWS CodeStar Connections

In this example, you want to grant access to the day-to-day usage of AWS CodeStar connections, such as creating and viewing details of connections, but not to more destructive actions, such as deleting connections.

```json
{

}
```
Example: A read-only-level policy for using AWS CodeStar Connections

In this example, you want to grant an IAM user in your account read-only access to the connections in your AWS account. This example shows how you might create a policy that allows viewing these items.

```json
{
  "Version": "2012-10-17",
  "Id": "Connections__ReadOnly",
  "Statement": [
    {
      "Sid": "Reads_API_Access",
      "Effect": "Allow",
      "Action": [
        "codestar-connections:GetConnection",
        "codestar-connections:ListConnections",
        "codestar-connections:ListInstallationTargets",
        "codestar-connections:GetInstallationUrl",
        "codestar-connections:GetIndividualAccessToken",
        "codestar-connections:StartOAuthHandshake",
        "codestar-connections:UpdateConnectionInstallation",
        "codestar-connections:ListTagsForResource"
      ],
      "Resource": "*"
    }
  ]
}
```

Example: A scoped-down policy for using AWS CodeStar Connections with a specified repository

In the following example, the customer wants the CodeBuild service role to access the specified Bitbucket repository. The policy on the CodeBuild service role:

```json
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "codestar-connections:UseConnection"
    ],
    "Condition": {
      "ForAllValues:StringEquals": {
        "codestar-connections:FullRepositoryId": "myrepoowner/myreponame"
      }
    }
  }
}
```
Example: A policy to use a connection with CodePipeline

In the following example, an administrator wants users to use a connection with CodePipeline. The policy attached to the user:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codestar-connections:PassConnection"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "codestar-connections:PassedToService": "codepipeline.amazonaws.com"
        }
      }
    }
  ]
}
```

Example: Use a CodeBuild service role for Bitbucket read operations with AWS CodeStar Connections

In the following example, the customer wants the CodeBuild service role to perform read operations on Bitbucket regardless of the repository. The policy on the CodeBuild service role:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codestar-connections:UseConnection"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "codestar-connections:ProviderPermissionsRequired": "read_only"
        }
      }
    }
  ]
}
```

Example: Limit the CodeBuild service role from performing operations with AWS CodeStar Connections

In the following example, the customer wants to prevent the CodeBuild service role from performing an operation like CreateRepository. The policy on the CodeBuild service role:

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codestar-connections:UseConnection"
      ],
      "Condition": {
        "ForAllValues:StringNotEquals": {
          "codestar-connections:ProviderPermissionsRequired": "CreateRepository"
        }
      }
    }
  ]
}
```
Using tags to control access to AWS CodeStar Connections resources

Tags can be attached to the resource or passed in the request to services that support tagging. In AWS CodeStar connections, resources can have tags, and some actions can include tags. When you create an IAM policy, you can use tag condition keys to control the following:

- Which users can perform actions on a pipeline resource, based on tags that it already has.
- Which tags can be passed in an action's request.
- Whether specific tag keys can be used in a request.

The following examples demonstrate how to specify tag conditions in policies for AWS CodeStar connections users.

Example 1: Allow actions based on tags in the request

The following policy grants users permission to create connections in AWS CodeStar connections.

To do that, it allows the CreateConnection and TagResource actions if the request specifies a tag named Project with the value ProjectA. (The aws:RequestTag condition key is used to control which tags can be passed in an IAM request.) The aws:TagKeys condition ensures tag key case sensitivity.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codestar-connections:CreateConnection",
        "codestar-connections:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/Project": "ProjectA"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["Project"]
        }
      }
    }
  ]
}
```

Example 2: Allow actions based on resource tags

The following policy grants users permission to perform actions on, and get information about, resources in AWS CodeStar connections.

To do that, it allows specific actions if the pipeline has a tag named Project with the value ProjectA. (The aws:RequestTag condition key is used to control which tags can be passed in an IAM request.) The aws:TagKeys condition ensures tag key case sensitivity.
Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete AWS CodeStar Notifications and AWS CodeStar Connections 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 using AWS managed policies** – To start using AWS CodeStar Notifications and AWS CodeStar Connections quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get started using permissions with AWS managed policies in the IAM User Guide.

- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant least privilege in the IAM User Guide.

- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using multi-factor authentication (MFA) in AWS in the IAM User Guide.

- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON policy elements: Condition in the IAM User Guide.

Using notifications and connections in the console

The notifications experience is built into the CodeBuild, CodeCommit, CodeDeploy, and CodePipeline consoles, as well as in the Developer Tools console in the Settings navigation bar itself. To access notifications in the consoles, you must either have one of the managed policies for those services applied, or you must have a minimum set of permissions. These permissions must allow you to list and view details about the AWS CodeStar Notifications and AWS CodeStar Connections 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 (IAM users or roles) with that policy. For more information about granting access to AWS CodeBuild, AWS CodeCommit, AWS CodeDeploy, and AWS CodePipeline, including access to those consoles, see the following topics:

- CodeBuild: Using identity-based policies for CodeBuild
- CodeCommit: Using identity-based policies for CodeCommit
- AWS CodeDeploy: Identity and access management for AWS CodeDeploy
- CodePipeline: Access control with IAM policies

AWS CodeStar Notifications does not have any AWS managed policies. To provide access to notification functionality, you must either apply one of the managed policies for one of the services listed previously, or you must create policies with the level of permission you want to grant to users or entities, and then attach those policies to the users, groups, or roles that require those permissions. For more information and examples, see the following:

- Example: An administrator-level policy for managing AWS CodeStar Notifications (p. 118)
- Example: A contributor-level policy for using AWS CodeStar Notifications (p. 119)
- Example: A read-only-level policy for using AWS CodeStar Notifications (p. 119).

AWS CodeStar Connections does not have any AWS managed policies. You use the permissions and combinations of permissions for access, such as the permissions detailed in Permissions for completing connections (p. 108).

For more information, see the following:

- Example: An administrator-level policy for managing AWS CodeStar Connections (p. 121)
- Example: A contributor-level policy for using AWS CodeStar Connections (p. 121)
- Example: A read-only-level policy for using AWS CodeStar Connections (p. 122)

You don’t need to allow 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 you’re trying to perform.

**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.

```json
{
   "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}"],
     },
   ]
}
```
Troubleshooting AWS CodeStar Notifications and AWS CodeStar Connections identity and access

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

Topics
- I want to view my access keys (p. 127)
- I'm an administrator and want to allow others to access notifications (p. 127)
- I created an Amazon SNS topic and added it as a notification rule target, but I am not receiving emails about events (p. 128)
- I want to allow people outside of my AWS account to access my AWS CodeStar Notifications and AWS CodeStar Connections resources (p. 128)

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, AKIAIOSFODNN7EXAMPLE) and a secret access key (for example, wJalrXUtnFEMI/K7MDENG/bPxRfi飘CYEXAMPLEKEY). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

**Important**
Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing access keys in the IAM User Guide.

I'm an administrator and want to allow others to access notifications

To allow others to access AWS CodeStar Notifications and AWS CodeStar Connections, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials
for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in AWS CodeStar Notifications and AWS CodeStar Connections.

To get started right away, see Creating your first IAM delegated user and group in the IAM User Guide.

For AWS CodeStar Notifications specific information, see Permissions and examples for AWS CodeStar Notifications (p. 115).

I created an Amazon SNS topic and added it as a notification rule target, but I am not receiving emails about events

In order to receive notifications about events, you must have a valid Amazon SNS topic subscribed as a target for the notification rule, and your email address must be subscribed to the Amazon SNS topic. To troubleshoot problems with the Amazon SNS topic, check the following:

- Make sure that the Amazon SNS topic is in the same AWS Region as the notification rule.
- Check to make sure that your email alias is subscribed to the correct topic, and that you have confirmed the subscription. For more information, see Subscribing an endpoint to an Amazon SNS topic.
- Verify that the topic policy has been modified to allow AWS CodeStar Notifications to push notifications to that topic. The topic policy should include a statement similar to the following:

```
{
   "Sid": "AWSCodeStarNotifications_publish",
   "Effect": "Allow",
   "Principal": {
      "Service": ["codestar-notifications.amazonaws.com"
   ],
   "Action": "SNS:Publish",
   "Condition": {
      "StringEquals": {
         "aws:SourceAccount": "123456789012"
      }
   }
}
```

For more information, see Setting up (p. 9).

I want to allow people outside of my AWS account to access my AWS CodeStar Notifications and AWS CodeStar Connections 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 CodeStar Notifications and AWS CodeStar Connections supports these features, see How features in the developer tools console work with IAM (p. 101).
- 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.

Using service-linked roles for AWS CodeStar Notifications

AWS CodeStar Notifications uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to AWS CodeStar Notifications. Service-linked roles are predefined by AWS CodeStar Notifications and include all the permissions that the service requires to call other AWS services on your behalf. This role is created for you the first time you create a notification rule. You don't have to create the role.

A service-linked role makes setting up AWS CodeStar Notifications easier because you don't have to add permissions manually. AWS CodeStar Notifications defines the permissions of its service-linked roles, and unless defined otherwise, only AWS CodeStar Notifications can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

To delete a service-linked role, you must first delete its related resources. This protects your AWS CodeStar Notifications resources because you can’t inadvertently remove permission to access the resources.

For information about other services that support service-linked roles, see AWS Services That Work with IAM.

Service-linked role permissions for AWS CodeStar Notifications

AWS CodeStar Notifications uses the AWSServiceRoleForCodeStarNotifications service-linked role to retrieve information about events that occur in your toolchain and send notifications to the targets you specify.

The AWSServiceRoleForCodeStarNotifications service-linked role trusts the following services to assume the role:

• codestar-notifications.amazonaws.com

The role permissions policy allows AWS CodeStar Notifications to complete the following actions on the specified resources:

• Action: PutRule on CloudWatch Event rules that are named awscodestar-notifications-*

• Action: DescribeRule on CloudWatch Event rules that are named awscodestar-notifications-*

• Action: PutTargets on CloudWatch Event rules that are named awscodestar-notifications-*

• Action: CreateTopic to create Amazon SNS topics for use with AWS CodeStar Notifications with the prefix CodeStarNotifications-

• Action: GetCommentsForPullRequests on all comments on all pull requests in all CodeCommit repositories in the AWS account
• Action: GetCommentsForComparedCommit on all comments on all commits in all CodeCommit repositories in the AWS account
• Action: GetDifferences on all commits in all CodeCommit repositories in the AWS account
• Action: GetCommentsForComparedCommit on all comments on all commits in all CodeCommit repositories in the AWS account
• Action: GetDifferences on all commits in all CodeCommit repositories in the AWS account
• Action: DescribeSlackChannelConfigurations on all AWS Chatbot clients in the AWS account
• Action: UpdateSlackChannelConfiguration on all AWS Chatbot clients in the AWS account
• Action: ListActionExecutions on all actions in all pipelines in the AWS account
• Action: GetFile on all files in all CodeCommit repositories in the AWS account unless otherwise tagged

You can see these actions in the policy statement for the AWSServiceRoleForCodeStarNotifications service-linked role.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "events:PutTargets",
        "events:PutRule",
        "events:DescribeRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/awscodestarnotifications-*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "sns:CreateTopic"
      ],
      "Resource": "arn:aws:sns:*:*:CodeStarNotifications-*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "codecommit:GetCommentsForPullRequest",
        "codecommit:GetCommentsForComparedCommit",
        "codecommit:GetDifferences",
        "chatbot:DescribeSlackChannelConfigurations",
        "chatbot:UpdateSlackChannelConfiguration",
        "codepipeline:ListActionExecutions"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "codecommit:GetFile"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/ExcludeFileContentFromNotifications": "true"
        }
      }
    }
  ]
}
```
You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the IAM User Guide.

Creating a service-linked role for AWS CodeStar Notifications

You don't need to manually create a service-linked role. You can use the Developer Tools console or the CreateNotificationRule API from the AWS CLI or SDKs to create a notification rule. You can also directly call the API. No matter which method you use, the service-linked role is created for you.

If you delete this service-linked role, and then need to create it again, you can use the same process to recreate the role in your account. You can use the Developer Tools console or the CreateNotificationRule API from the AWS CLI or SDKs to create a notification rule. You can also directly call the API. No matter which method you use, the service-linked role is created for you.

Editing a service-linked role for AWS CodeStar Notifications

After you create a service-linked role, you cannot change its name because various entities might reference the role. However, you can use IAM to edit the role description. For more information, see Editing a Service-Linked Role in the IAM User Guide.

Deleting a service-linked role for AWS CodeStar Notifications

If you no longer need to use a feature or service that requires a service-linked role, we recommend that you delete the role. That way, you don't have an unused entity that is not actively monitored or maintained. You must clean up the resources for your service-linked role before you can delete it. For AWS CodeStar Notifications, this means deleting all notification rules that use the service role in your AWS account.

Note
If the AWS CodeStar Notifications service is using the role when you try to delete the resources, then the deletion might fail. If that happens, wait for a few minutes and try the operation again.

To delete AWS CodeStar Notifications resources used by AWSServiceRoleForCodeStarNotifications

   
   Note
   Notification rules apply to the AWS Region where they are created. If you have notification rules in more than one AWS Region, use the Region selector to change the AWS Region.

2. Choose all notification rules that appear in the list, and then choose Delete.

3. Repeat these steps in all AWS Regions where you created notification rules.

To use IAM to delete the service-linked role

Use the IAM console, AWS CLI, or AWS Identity and Access Management API to delete the AWSServiceRoleForCodeStarNotifications service-linked role. For more information, see Deleting a Service-Linked Role in the IAM User Guide.
Supported regions for AWS CodeStar Notifications service-linked roles

AWS CodeStar Notifications supports using service-linked roles in all of the AWS Regions where the service is available. For more information, see AWS Regions and Endpoints and AWS CodeStar Notifications.

Compliance validation for AWS CodeStar Notifications and AWS CodeStar Connections

AWS CodeStar Notifications and AWS CodeStar Connections are not in scope of any AWS compliance programs.

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 CodeStar Notifications and AWS CodeStar Connections 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.
- AWS compliance resources – This collection of workbooks and guides might apply to your industry and location.
- AWS Config – This AWS service 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 CodeStar Notifications and AWS CodeStar Connections

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability 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.

- Notification rules are specific to the AWS Region where they are created. If you have notification rules in more than one AWS Region, use the Region selector to review notification rules in each AWS Region.
- AWS CodeStar Notifications relies on Amazon Simple Notification Service (Amazon SNS) topics as notification rule targets. Information about your Amazon SNS topics and notification rule targets
might be stored in an AWS Region different from the Region in which you configured the notification rule.

Infrastructure security in AWS CodeStar Notifications and AWS CodeStar Connections

As features in a managed service, AWS CodeStar Notifications and AWS CodeStar Connections are 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 CodeStar Notifications and AWS CodeStar Connections through the network. Clients must support Transport Layer Security (TLS) 1.0 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 support these modes.

Requests must be signed by 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.

Traffic between AWS CodeStar Connections resources across regions

If you use the connections feature to enable connection of your resources, you agree and instruct us to store and process information associated with such connection resources in AWS Regions outside the AWS Regions where you are using the underlying service, solely in connection with, and for the sole purpose of, providing connection to such resources in Regions other than the one where the resource was created.

For more information, see Global resources in AWS CodeStar Connections (p. 41).
Document history

The following table describes the documentation for this release of the Developer Tools console.

- **AWS CodeStar Notifications API version**: 2019-10-15
- **AWS CodeStar Connections API version**: 2019-12-01
- **Latest documentation update**: September 02, 2021

<table>
<thead>
<tr>
<th>update-history-change</th>
<th>update-history-description</th>
<th>update-history-date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Added information for tagging host resources (p. 134)</td>
<td>You can now tag hosts using the console and the CLI. For more information, see Tag resources in AWS CodeStar Connections.</td>
<td>April 19, 2021</td>
</tr>
<tr>
<td>VPC endpoint support for connections (p. 134)</td>
<td>You can now use VPC endpoints with connections. For more information, see AWS CodeStar Connections and interface VPC endpoints (AWS PrivateLink).</td>
<td>November 24, 2020</td>
</tr>
<tr>
<td>New GitHub and GitHub Enterprise Cloud provider types (p. 134)</td>
<td>You can now create connections to GitHub and GitHub Enterprise Cloud. For more information, see Create a connection and Create a connection to GitHub.</td>
<td>September 30, 2020</td>
</tr>
<tr>
<td>Added the GitHub Enterprise Server provider type and host resources (p. 134)</td>
<td>Information about the host resource for connections has been added to this guide. You can now create connections to GitHub Enterprise Server. For more information, see Create a connection and Working with hosts. This is the general availability release of the connections feature in the Developer Tools console User Guide.</td>
<td>June 29, 2020</td>
</tr>
<tr>
<td>Added information for using and tagging connections (p. 134)</td>
<td>Information about the connections feature in the console has been added to this guide. You can view concepts, steps for getting started, a permissions reference including example policies, and steps to create, view, and tag connections. For more information, see What are connections, Connections concepts, Getting started with connections, Create a</td>
<td>June 28, 2020</td>
</tr>
</tbody>
</table>
connection, Tag resources in AWS CodeStar Connections, Security, Quotas for connections, Troubleshooting, and AWS CodeStar Connections API calls with AWS CloudTrail. To view a list of additional provider actions (permissions only actions), see Actions for ProviderType.

New target type for notification rules (p. 134)

You can now choose AWS Chatbot clients configured for Slack channels as the target for notification rules. For more information, see Create a notification rule and Working with notification rule targets.

April 2, 2020

Added notifications about additional AWS CodeCommit events (p. 134)

You can now configure notifications for events related to pull request approvals. For more information, see Events for notification rules on repositories and Working with pull requests in CodeCommit.

February 10, 2020

Notifications available in two additional AWS regions (p. 134)

The Developer Tools console now supports notifications in Middle East (Bahrain) and Asia Pacific (Hong Kong). For more information, see AWS CodeStar Notifications in the AWS General Reference.

February 5, 2020

Added support for encrypted Amazon SNS topics (p. 134)

Guidance has been added for using encrypted Amazon SNS topics as notification targets. For more information, see Configure Amazon SNS topics for notifications.

February 4, 2020

Notifications can include session tag information for CodeCommit (p. 134)

Notifications for CodeCommit can now contain user identity information, such as a display name or an email address, through the use of session tags. For more information, see Concepts and Using tags to provide identity information in CodeCommit.

December 19, 2019

Initial release (p. 134)

This is the initial release of the Developer Tools console User Guide.

November 5, 2019
AWS glossary

For the latest AWS terminology, see the AWS glossary in the AWS General Reference.