Reference Guide

AWS Account Management

AWS Account Management: Reference Guide

Table of Contents

Welcome to the AWS Account Management Reference Guide

AWS accounts are a fundamental part of accessing AWS services.

An AWS account serves two basic functions:

- **Container** – An AWS account is the basic container for all the AWS resources you create as an AWS customer. For example, an Amazon Simple Storage Service (Amazon S3) bucket, an Amazon Relational Database Service (Amazon RDS) database, and an Amazon Elastic Compute Cloud (Amazon EC2) instance are all resources. Every resource is uniquely identified by an Amazon Resource Name (ARN) that includes the account ID of the account that contains, or owns, the resource.
- **Security boundary** – An AWS account is also the basic security boundary for your AWS resources. Resources that you create in your account are available to users who have credentials for your account.

  Among the key resources you can create in your account are *identities*, such as users and roles. Identities have credentials that someone can use to sign in (*authenticate*) to AWS. Identities also have permission policies that specify what a user can do (*authorization*) with the resources in the account.

  As a security best practice, require your users to use temporary credentials when accessing AWS. To provide temporary credentials, you can use federation and an identity provider, such as AWS IAM Identity Center (IAM Identity Center). If your company already uses an identity provider, use it with federation to simplify how you provide access to the resources in your AWS account.

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

Topics

Do I need multiple AWS accounts?

AWS accounts serve as the fundamental security boundary in AWS. They serve as a resource container that provides a useful level of isolation. The ability to isolate resources and users is a key requirement to establishing a secure, well governed environment.

Separating your resources into separate AWS accounts helps you to support the following principles in your cloud environment:

- **Security control** – Different applications can have different security profiles, requiring different control policies and mechanisms around them. For example, it's far easier to talk to an auditor and be able to point to a single AWS account that hosts all elements of your workload that are subject to Payment Card Industry (PCI) Security Standards.
- **Isolation** – An AWS account is a unit of security protection. Potential risks and security threats should be contained within an AWS account without affecting others. There could be different security needs due to different teams or different security profiles.
- **Many teams** – Different teams have their different responsibilities and resource needs. You can prevent teams from interfering with each other by moving them to separate AWS accounts.
- **Data isolation** – In addition to isolating the teams, it's important to isolate the data stores to an account. This can help limit the number of people that can access and manage that data store. This helps contain exposure to highly private data and therefore can help in compliance with the European Union's General Data Protection Regulation (GDPR).
- **Business process** – Different business units or products may have completely different purposes and processes. With multiple AWS accounts, you can support a business unit's specific needs.
- **Billing** – An account is the only true way to separate items at a billing level. Multiple accounts help separate items at a billing level across business units, functional teams, or individual users. You can still get all of your bills consolidated to a single payer (using AWS Organizations and consolidated billing) while having line items separated by AWS account.
- **Quota allocation** – AWS service quotas are enforced separately for each AWS account. Separating workloads into different AWS accounts prevents them from consuming quotas for each other.

All of the recommendations and procedures described in this document are in compliance with the AWS Well-Architected Framework. This framework is intended to help you design a flexible, resilient, and scalable cloud infrastructure. Even when you are starting small, we recommend that

you proceed in compliance with this guidance in the framework. Doing so can help you scale your environment securely and without impacting your ongoing operations as you grow.

Managing multiple AWS accounts

Before you start adding multiple accounts, you'll want to develop a plan to manage them. For that, we recommend that you use AWS Organizations, which is a free AWS service to manage all of the AWS accounts in your organization.

AWS also offers AWS Control Tower, which adds layers of AWS managed automation to Organizations and automatically integrates it with other AWS services like AWS CloudTrail, AWS Config, Amazon CloudWatch, AWS Service Catalog, and others. These services can incur additional costs. For more information, see AWS Control Tower pricing.

Getting started: Are you a first-time AWS user?

If you're a first-time user of AWS, your first step is to sign up for an AWS account. When you sign up, AWS creates an AWS account with the details that you provide and assigns the account to you. After you create your AWS account, sign in as the root user, activate multi-factor authentication (MFA) for the root user, and assign administrative access to a user.

Steps

Prerequisites

To sign up for an AWS account, you need the following information:

- **An account name** – The name of the account appears in several places, such as on your invoice, and in consoles such as the Billing and Cost Management dashboard and the AWS Organizations console.

  We recommend that you use a standard way to name your accounts so that you can give your accounts names that are easy to recognize. For company accounts, consider using a naming

standard such as *organization-purpose-environment* (for example, *AnyCompany-audit-prod*). For personal accounts, consider using a naming standard such as *first name-last name-purpose* (for example, *paulo-santos-testaccount*).

For information about changing an account name, see How do I change the name on my AWS account?.

- **Address** – If your contact address is in India, the user agreement for your account is with Amazon Internet Services Private Limited (AISPL), a local AWS seller in India. You must provide your CVV as part of the verification process. You might also have to enter a one-time password, depending on your bank. AISPL charges your payment method 2 INR as part of the verification process. AISPL refunds the 2 INR after verification is complete.
- **An email address** – The email address is used as the sign-in name for the root user and is required for account recovery. You must be able to receive email messages that are sent to this address. Before you can perform certain tasks, you must verify that you have access to email sent to this address.

  > **Important**
  >
  > If this account is for a business, use a secure corporate distribution list (for example, `it.admins@example.com`) so that your company can retain access to the AWS account even when an employee changes positions or leaves the company. Because the email address can be used to reset the account's root user credentials, protect access to this distribution list or address.

- **A phone number** – This number can be used to confirm the ownership of your account. You must be able to receive calls at this phone number.

  > **Important**
  >
  > If this account is for a business, use a corporate phone number so that your company can retain access to the AWS account even when an employee changes positions or leaves the company.

Step 1: Create your AWS account

1. In your browser, open the AWS home page.

2. Choose **Create an AWS account**.

> **Note**
>
> If you signed in to AWS recently, choose **Sign in**. If the option **Create a new AWS account** isn't visible, first choose **Sign in to a different account**, and then choose **Create a new AWS account**.

3. Enter your account information, and then choose **Verify email address**. This will send a verification code to your specified email address.
4. Enter your verification code, and then choose **Verify**.
5. Enter a strong password for your root user, confirm it, and then choose **Continue**. AWS requires that your password meet the following conditions:
   - It must have a minimum of 8 characters and a maximum of 128 characters.
   - It must include a minimum of three of the following mix of character types: uppercase, lowercase, numbers, and ! @ # $ % ^ & * () <> [] {} | _+-= symbols.
   - It must not be identical to your AWS account name or email address.
6. Choose **Business** or **Personal**. The difference between these options is the information that we ask you for. Both account types have the same features and functions.
7. Enter your business or personal information. Refer to the recommendations in the Prerequisites section about the email address and phone number.
8. Read and accept the AWS Customer Agreement. Be sure that you read and understand the terms of the AWS Customer Agreement.
9. Choose **Continue**. At this point, you'll receive an email message to confirm that your AWS account is ready to use. You can sign in to your new account by using the email address and password you provided during sign up. However, you can't use any AWS services until you finish activating your account.
10. Enter information about your payment method. If you want to use a different address for billing purposes, choose **Use a new address**.
11. Choose **Verify and Continue**.
12. Enter your country or region code from the list, and then enter a phone number where you can be reached in the next few minutes. Enter the CAPTCHA code, and submit.
13. When the automated system contacts you, enter the PIN you receive and then submit.

14. Select your AWS Support plan. For a description of the available plans, see Compare AWS Support plans.
15. Choose **Complete sign up**. A confirmation page appears that indicates that your account is being activated.
16. Check your email and spam folder for an email message that confirms your account was activated. Activation usually takes a few minutes but can sometimes take up to 24 hours.

    After you receive the activation message, you have full access to all AWS services.

> **Note**
>
> If you are having trouble with account activation, see the section called "Account creation issues".

Step 2: Activate MFA for your root user

We strongly recommend that you activate MFA for your root user. MFA dramatically lowers the risk of someone accessing your account without your authorization.

1. Sign in to the AWS Management Console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   For help signing in using your root user, see Sign in to the AWS Management Console as the root user in the *AWS Sign-In User Guide*.
2. Turn on MFA for your root user.

   For instructions, see Enable a virtual MFA device for your AWS account root user (console) in the *IAM User Guide*.

Step 3: Create an administrator user

Because you can't restrict what a root user can do, we strongly recommend that you don't use your root user for any tasks that don't explicitly require the root user. Instead, assign administrative access to an administrative user in IAM Identity Center, and sign in as that administrative user to perform your daily administrative tasks.

For instructions, see Set up AWS account access for an IAM Identity Center administrative user in the *IAM Identity Center User Guide*.

Related topics

- For information about protecting your root user credentials, see Securing the credentials for the root user in the *IAM User Guide*.
- For a list of tasks that require the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Using the AWS account root user

> **Important**
>
> Anyone who has root user credentials for your AWS account has unrestricted access to all the resources in your account, including billing information.

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

To avoid using the root user for everyday tasks, learn how to set up an administrative user in AWS IAM Identity Center. For additional root user security recommendations, see Root user best practices for your AWS account.

You can change, or reset the root user password, and create, or delete access keys (access key IDs and secret access keys) for your root user. For help signing in using your root user, see Sign in to the AWS Management Console as the root user in the *AWS Sign-In User Guide*.

Manage your AWS account

This section includes topics that describe how to manage your AWS account.

> **Note**
>
> If your AWS account was created in India by using Amazon Internet Services Private Limited (AISPL), there are additional considerations. For more information, see Manage accounts in India.

Topics

Create a standalone AWS account

This topic describes how to create a standalone AWS account that isn't managed by AWS Organizations. If you want to create an account that's part of an organization managed by AWS Organizations, see Creating a member account in your organization in the *AWS Organizations User Guide*.

These instructions are for creating an AWS account outside of India. For creating an account in India, see Create an AWS account with AISPL.

AWS Management Console

To create an AWS account

1. Open the Amazon Web Services home page .
2. Choose **Create an AWS account**.

> **Note**
>
> If you signed in to AWS recently, that option might not be there. Instead, choose **Sign in to the Console**. Then, if **Create a new AWS account** still isn't visible, first choose **Sign in to a different account**, and then choose **Create a new AWS account**.

3. Enter your account information, and then choose **Verify email address**. This will send a verification code to your specified email address.

> **Important**
>
> Because of the critical nature of the root user of the account, we strongly recommend that you use an email address that can be accessed by a group, rather than only an individual. That way, if the person who signed up for the AWS account leaves the company, the AWS account can still be used because the email address is still accessible.
> If you lose access to the email address associated with the AWS account, then you can't recover access to the account if you ever lose the password.

4. Enter your verification code, and then choose **Verify**.
5. Enter a strong password for your root user, confirm it, and then choose **Continue**. AWS requires that your password meet the following conditions:
   - It must have a minimum of 8 characters and a maximum of 128 characters.
   - It must include a minimum of three of the following mix of character types: uppercase, lowercase, numbers, and ! @ # $ % ^ & * () <> [] {} | _+-= symbols.
   - It must not be identical to your AWS account name or email address.
6. Choose **Business** or **Personal**. Personal accounts and business accounts have the same features and functions.
7. Enter your company or personal information.

> **Important**
>
> For business AWS accounts, it's a best practice to enter:
>
> - A company phone number rather than a number for a personal phone.
> - An e-mail address with a domain name that belongs to the company or organization that will be using the account.
>
> Configuring the account's root user with an individual email address or a personal phone number can make your account insecure.

8. Read and accept the AWS Customer Agreement. Be sure that you read and understand the terms of the AWS Customer Agreement.
9. Choose **Continue**. At this point, you'll receive an email message to confirm that your AWS account is ready to use. You can sign in to your new account by using the email address and password you provided during sign up. However, you can't use any AWS services until you finish activating your account.
10. Enter the information about your payment method, and then choose **Verify and Continue**. If you want to use a different billing address for your AWS billing information, choose **Use a new address**.

    You can't proceed with the sign-up process until you add a valid payment method.
11. Enter your country or region code from the list, and then enter a phone number where you can be reached in the next few minutes.
12. Enter the code displayed in the CAPTCHA, and then submit.
13. When the automated system contacts you, enter the PIN you receive and then submit.
14. Select one of the available AWS Support plans. For a description of the available Support plans and their benefits, see Compare AWS Support plans.
15. Choose **Complete sign up**. A confirmation page appears that indicates that your account is being activated.
16. Check your email and spam folder for an email message that confirms your account was activated. Activation usually takes a few minutes but can sometimes take up to 24 hours.

    After you receive the activation message, you have full access to all AWS services.

AWS CLI & SDKs

You can create member accounts in an organization that is managed by AWS Organizations by running the CreateAccount operation while signed in to the organization's management account.

You can't create a standalone AWS account outside of an organization by using an AWS Command Line Interface (AWS CLI) or AWS API operation.

View AWS account identifiers

AWS assigns the following unique identifiers to each AWS account:

AWS account ID

A 12-digit number, such as 012345678901, that uniquely identifies an AWS account. Many AWS resources include the account ID in their Amazon Resource Names (ARNs). The account ID portion distinguishes resources in one account from the resources in another account. If you're an AWS Identity and Access Management (IAM) user, you can sign in to the AWS Management Console using either the account ID or account alias. While account IDs, like any identifying information, should be used and shared carefully, they are not considered secret, sensitive, or confidential information.

Canonical user ID

An alpha-numeric identifier, such as `79a59df900b949e55d96a1e698fbacedfd6e09d98eacf8f8d5218e7cd47ef2be`, that is an obfuscated form of the AWS account ID. You can use this ID to identify an AWS account when granting cross-account access to buckets and objects using Amazon Simple Storage Service (Amazon S3). You can retrieve the canonical user ID for your AWS account as either the root user or an IAM user.

You must be authenticated with AWS to view these identifiers.

> **Warning**
>
> Do not provide your AWS credentials (including passwords and access keys) to a third party that needs your AWS account identifiers to share AWS resources with you. Doing so would give them the same access to the AWS account that you have.

Find your AWS account ID

You can find the AWS account ID using either the AWS Management Console or the AWS Command Line Interface (AWS CLI). In the console, the location of the account ID depends on whether you're signed in as the root user or an IAM user. The account ID is the same whether you're signed in as the root user or an IAM user.

Finding your account ID as the root user

AWS Management Console

To find your AWS account ID when signed in as the root user

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:
>
> - When you sign in as the root user, you don't need any IAM permissions.

1. In the navigation bar on the upper right, choose your account name or number and then choose **Security credentials**.

   > **Tip**
   >
   > If you don't see the **Security credentials** option, you might be signed in as a federated user with an IAM role, instead of as an IAM user. In this case, look for the entry **Account** and the account ID number next to it.

2. Under the **Account details** section, the account number appears next to **AWS account ID**.

AWS CLI & SDKs

To find your AWS account ID using the AWS CLI

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:

- When you run the command as the root user, you don't need any IAM permissions.

Use the get-caller-identity command as follows.

```
$ aws sts get-caller-identity \
    --query Account \
    --output text
123456789012
```

Find your account ID as an IAM user

AWS Management Console

To find your AWS account ID when signed in as an IAM user

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:
>
> - `account:GetAccountInformation`

1. In the navigation bar on the upper right, choose your user name and then choose **Security credentials**.

   > **Tip**
   >
   > If you don't see the **Security credentials** option, you might be signed in as a federated user with an IAM role, instead of as an IAM user. In this case, look for the entry **Account** and the account ID number next to it.

2. At the top of the page, under **Account details**, the account number appears next to **AWS account ID**.

AWS CLI & SDKs

To find your AWS account ID using the AWS CLI

Minimum permissions

To perform the following steps, you must have at least the following IAM permissions:

- When you run the command as an IAM user or role, then you must have:
  - `sts:GetCallerIdentity`

Use the get-caller-identity command as follows.

```
$ aws sts get-caller-identity \
    --query Account \
    --output text
123456789012
```

Find the canonical user ID for your AWS account

You can find the canonical user ID for your AWS account using the AWS Management Console or the AWS CLI. The canonical user ID for an AWS account is specific to that account. You can retrieve the canonical user ID for your AWS account as the root user, a federated user, or an IAM user.

Find the canonical ID as the root user or IAM user

AWS Management Console

To find the canonical user ID for your account when signed in to the console as the root user or an IAM user

Minimum permissions

To perform the following steps, you must have at least the following IAM permissions:

- When you run the command as the root user, you don't need any IAM permissions.
- When you sign in as an IAM user, then you must have:
  - `account:GetAccountInformation`

1. Sign in to the AWS Management Console as the root user or an IAM user.

2. In the navigation bar on the upper right, choose your account name or number and then choose **Security credentials**.

   > **Tip**
   >
   > If you don't see the **Security credentials** option, you might be signed in as a federated user with an IAM role, instead of as an IAM user. In this case, look for the entry **Account** and the account ID number next to it.

3. Under the **Account details** section, the canonical user ID appears next to **Canonical user ID**. You can use your canonical user ID to configure Amazon S3 access control lists (ACLs).

AWS CLI & SDKs

To find the canonical user ID using the AWS CLI

The same AWS CLI and API command works for the AWS account root user, IAM users, or IAM roles.

Use the list-buckets command as follows.

```
$ aws s3api list-buckets \
    --query Owner.ID \
    --output text
249fa2f1dc32c330EXAMPLE91b2778fcc65f980f9172f9cb9a5f50ccbEXAMPLE
```

Find the canonical ID as a federated user with an IAM role

AWS Management Console

To find the canonical ID for your account when signed in to the console as a federated user with an IAM role

> **Minimum permissions**
>
> - You must have permission to list and view an Amazon S3 bucket.

1. Sign in to the AWS Management Console as a federated user with an IAM role.

2. In the Amazon S3 console, choose a bucket name to view details about a bucket.
3. Choose the **Permissions** tab.
4. In the **Access control list** section, under **Bucket owner**, the canonical ID for your AWS account appears.

AWS CLI & SDKs

To find the canonical user ID using the AWS CLI

The same AWS CLI and API command works for the AWS account root user, IAM users, or IAM roles.

Use the list-buckets command as follows.

```
$ aws s3api list-buckets \
    --query Owner.ID \
    --output text
249fa2f1dc32c330EXAMPLE91b2778fcc65f980f9172f9cb9a5f50ccbEXAMPLE
```

Update the AWS account name, email address, or password for the root user

To edit your AWS account's name, or to change the root user's password or email address, perform the steps in the following procedure. This email address and password are the credentials you use to sign in as the AWS account root user.

> **Note**
>
> Changes to an AWS account can take up to four hours to propagate everywhere.

AWS Management Console

To edit your AWS account name, root user password, or root user email address

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:

- You must sign in as the AWS account root user, which requires no additional IAM permissions. You can't perform these steps as an IAM user or role.

1. Use your AWS account's email address and password to sign in to the AWS Management Console as your AWS account root user.
2. In the upper right corner of the console, choose your account name or number and then choose **Account**.
3. On the **Account** page, next to **Account settings**, choose **Edit**. You are prompted to re-authenticate for security purposes.

   Note

   If you don't see the **Edit** option, it is likely that you are not signed in as the root user for your account. You can't modify account settings while signed in as an IAM user or role.

4. On the **Update account settings** page, choose **Edit** next to the field that you want to update.

   a. For **Name** – On the **Update your account name** page, in **New account name**, enter the new account name, and then choose **Save changes**.

      Note

      If you are unable to modify the AWS account name, check if a service control policy (SCP) exists in AWS Organizations that restricts access to `account` or is set to deny the `iam:UpdateAccountName` action.

   b. For **Email** – On the **Update your email address** page, fill out the fields for **New email address**, **Confirm new email address**, and confirm your current **Password**. Then, choose **Save changes**. A verification code is sent to your new email address from `no-reply@verify.signin.aws`. On the **Verify your new email address** page, under **Verification code**, enter the code you received from your email, and then choose **Save changes**.

> **Note**
>
> It can take up to 5 minutes for the verification code to arrive. If you don't see the email in your inbox, check your spam and junk folders.

c. For **Password** – On the **Update your password** page, fill out the fields for **Current password**, **New password**, and **Confirm new password**. Then, choose **Save changes**. For additional guidance including best practices for setting root user passwords, see Change the password for the AWS account root user in the *IAM User Guide*.

5. After you have made all of your changes, choose **Done**.

AWS CLI & SDKs

This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

Understanding API modes of operation

The API operations that work with an AWS account's attributes always work in one of two modes of operation:

- **Standalone context** – this mode is used when a user or role in an account accesses or changes an account attribute in the ***same account***. The standalone context mode is automatically used when you ***don't*** include the `AccountId` parameter when you call one of the Account Management AWS CLI or AWS SDK operations.
- **Organizations context** – this mode is used when a user or role in one account in an organization accesses or changes an account attribute in a different member account in the same organization. The organizations context mode is automatically used when you ***do*** include the `AccountId` parameter when you call one of the Account Management AWS CLI or AWS SDK operation. You can call the operations in this mode from only the management account of the organization, or the delegated admin account for Account Management.

The AWS CLI and AWS SDK operations can work in either standalone or organizations context.

- If you ***don't*** include the `AccountId` parameter, then the operation runs in the standalone context and automatically applies the request to the account you used to make the request. This is true whether or not the account is a member of an organization.
- If you do include the `AccountId` parameter, then the operation runs in the organizations context, and the operation works on the specified Organizations account.
  - If the account calling the operation is the management account or the delegated admin account for the Account Management service, then you can specify any member account of that organization in the `AccountId` parameter to update the specified account.
  - The only account in an organization that can call one of the alternate contact operations and specify its own account number in the `AccountId` parameter is the account specified as the delegated admin account for the Account Management service. Any other account, including the management account, receives an `AccessDenied` exception.
- If you run an operation in standalone mode, then you must be permitted to run the operation with an IAM policy that includes a `Resource` element of either `"*"` to allow all resources, or an ARN that uses the syntax for a standalone account.
- If you run an operation in organizations mode, then you must be permitted to run the operation with an IAM policy that includes a `Resource` element of either `"*"` to allow all resources, or an ARN that uses the syntax for a member account in an organization.

Granting permissions to update account attributes

As with most AWS operations, you grant permissions to add, update, or delete account attributes for AWS accounts by using IAM permission policies. When you attach an IAM permission policy to an IAM principal (either a user or role), you specify which actions that principal can perform on which resources, and under what conditions.

The following are some Account Management specific considerations for creating a permissions policy.

Amazon Resource Name format for AWS accounts

- The Amazon Resource Name (ARN) for an AWS account that you can include in the `resource` element of a policy statement is constructed differently based on whether the account you want to reference is a standalone account or an account that is in an organization. See the previous section on Understanding API modes of operation.
  -

An account ARN for a standalone account:

```
arn:aws:account::{AccountId}:account
```

You must use this format when you run an account attributes operation in standalone mode by not including the `AccountID` parameter.

- An account ARN for a member account in an organization:

  ```
  arn:aws:account::{ManagementAccountId}:account/o-{OrganizationId}/{AccountId}
  ```

  You must use this format when you run an account attributes operation in organizations mode by including the `AccountID` parameter.

Context keys for IAM policies

The Account Management service also provides several Account Management service-specific condition keys that provide fine-grained control over the permissions you grant.

`account:AccountResourceOrgPaths`

The context key `account:AccountResourceOrgPaths` lets you specify a path through your organization's hierarchy to a specific organizational unit (OU). Only member accounts that are contained by that OU match the condition. The following example snippet restricts the policy to apply to only accounts that are in either of two specified OUs.

Because `account:AccountResourceOrgPaths` is a multi-valued string type, you must use the `ForAnyValue` or `ForAllValues` multi-value string operators. Also, note that the prefix on the condition key is `account`, even though you're referencing paths to OUs in an organization.

```
"Condition": {
    "ForAnyValue:StringLike": {
        "account:AccountResourceOrgPaths": [
            "o-aa111bb222/r-a1b2/ou-a1b2-f6g7h111/*",
            "o-aa111bb222/r-a1b2/ou-a1b2-f6g7h222/*"
        ]
    }
}
```

`account:AccountResourceOrgTags`

The context key `account:AccountResourceOrgTags` lets you reference the tags that can be attached to an account in an organization. A tag is a key/value string pair that you can use to categorize and label the resources in your account. For more information about tagging, see Tag Editor in the *AWS Resource Groups User Guide*. For information about using tags as part of an attribute-based access control strategy, see What is ABAC for AWS in the *IAM User Guide*. The following example snippet restricts the policy to apply to only accounts in an organization that have the tag with the key `project` and a value of either `blue` or `red`.

Because `account:AccountResourceOrgTags` is a multi-valued string type, you must use the `ForAnyValue` or `ForAllValues` multi-value string operators. Also, note that the prefix on the condition key is `account`, even though you're referencing the tags on an organization's member account.

```
"Condition": {
    "ForAnyValue:StringLike": {
        "account:AccountResourceOrgTags/project": [
            "blue",
            "red"
        ]
    }
}
```

Note

You can attach tags to only an account in an organization. You can't attach tags to a standalone AWS account.

Update your AWS account contact information

You can store contact information about the primary account contact for your AWS account. You can also add or edit contact information for the following alternate account contacts:

- **Billing** – The alternate billing contact will receive billing-related notifications, such as invoice availability notifications.
- **Operations** – The alternate operations contact will receive operations-related notifications.

- **Security** – The alternate security contact will receive security-related notifications, including notifications from the AWS Abuse Team.

Topics

Update the alternate contacts for your AWS account

Alternate contacts allows AWS to contact up to three alternate contacts associated with the account. An alternate contact doesn't have to be a specific person. You could instead add an email distribution list if you have a team that manages billing, operations and security related issues. These are in addition to the email address associated with the root user of the account. The primary account contact will continue to receive all email communications sent to the root account's email.

You can specify only one of each of the following contact types associated with an account.

- Billing contact
- Operations contact
- Security contact

You can add or edit alternate contacts differently, depending on whether or not the accounts are standalone, or part of an organization:

- **Standalone AWS accounts** – For AWS accounts not associated with an organization, you can update your own alternate contacts using the AWS Management Console, or via AWS CLI & SDKs. To learn how to do this, see Update standalone AWS account alternate contacts.
- **AWS accounts within an organization** – For member accounts that are part of an AWS organization, a user in the management account or delegated admin account can centrally update any member account in the organization from the AWS Organizations console, or programmatically via the AWS CLI & SDKs. To learn how to do this, see Update AWS account alternate contacts in your organization.

Topics

- Phone number and email address requirements
- Update the alternate contacts for a standalone AWS account
- Update the alternate contacts for any AWS account in your organization
- account:AlternateContactTypes context key

Phone number and email address requirements

Before you proceed with updating your account's alternate contacts information, we recommend that you first review the following requirements when entering phone numbers and email addresses.

- Phone numbers can only contain numbers, whitespaces and the following characters:"+ - ( )".
- Email addresses can be up to 254 characters long and can include the following special characters in the local portion of the email address in addition to the standard alphanumeric ones: "+ = . # | ! & - _".

Update the alternate contacts for a standalone AWS account

To add or edit the alternate contacts for a standalone AWS account, perform the steps in the following procedure. The AWS Management Console procedure below always works *only* in the standalone context. You can use the AWS Management Console to access or change only the alternate contacts in the account you used to call the operation.

AWS Management Console

To add or edit the alternate contacts for a standalone AWS account

Minimum permissions

To perform the following steps, you must have at least the following IAM permissions:

- `account:GetAlternateContact` (to see the alternate contact details)
- `account:PutAlternateContact` (to set or update an alternate contact)
- `account:DeleteAlternateContact` (to delete an alternate contact)

1. Sign in to the AWS Management Console as an IAM user or role that has the minimum permissions.
2. Choose your account name on the top right of the window, and then choose **Account**.
3. On the **Account** page, scroll down to **Alternate contacts**, and to the right of the title, choose **Edit**.

> **Note**
>
> If you don't see the **Edit** option, it is likely that you are not signed in as the root user for your account or as someone who has the minimum permissions specified above.

4. Change the values in any of the available fields.

> **Important**
>
> For business AWS accounts, it's a best practice to enter a company phone number and email address rather than one belonging to an individual.

5. After you have made all of your changes, choose **Update**.

AWS CLI & SDKs

You can retrieve, update, or delete the ***alternate*** contact information by using the following AWS CLI commands or their AWS SDK equivalent operations:

- GetAlternateContact
- PutAlternateContact
- DeleteAlternateContact

> **Notes**
>
> - To perform these operations from the management account or a delegated admin account in an organization against member accounts, you must enable trusted access for the Account service.

Minimum permissions

For each operation, you must have the permission that maps to that operation:

- `GetAlternateContact` (to see the alternate contact details)
- `PutAlternateContact` (to set or update an alternate contact)
- `DeleteAlternateContact` (to delete an alternate contact)

If you use these individual permissions, you can grant some users the ability to only read the contact information, and grant others the ability to both read and write.

Example

The following example retrieves the current Billing alternate contact for the caller's account.

```
$ aws account get-alternate-contact \
    --alternate-contact-type=BILLING
{
    "AlternateContact": {
        "AlternateContactType": "BILLING",
        "EmailAddress": "saanvi.sarkar@amazon.com",
        "Name": "Saanvi Sarkar",
        "PhoneNumber": "+1(206)555-0123",
        "Title": "CFO"
    }
}
```

Example

The following example sets a new Operations alternate contact for the caller's account.

```
$ aws account put-alternate-contact \
    --alternate-contact-type=OPERATIONS \
    --email-address=mateo_jackson@amazon.com \
    --name="Mateo Jackson" \
    --phone-number="+1(206)555-1234" \
```

```
    --title="Operations Manager"
```

This command produces no output if it's successful.

Example

> **Note**
>
> If you perform multiple `PutAlternateContact` operations on the same AWS account and the same contact type, the first adds the new contact, and all successive calls to the same AWS account and contact type update the existing contact.

Example

The following example deletes the Security alternate contact for the caller's account.

```
$ aws account delete-alternate-contact \
    --alternate-contact-type=SECURITY
```

This command produces no output if it's successful.

> **Note**
>
> If you try to delete the same contact more than once, the first succeeds silently. All later attempts generate a `ResourceNotFound` exception.

Update the alternate contacts for any AWS account in your organization

To add or edit the alternate contact details for any AWS account in your organization, perform the steps in the following procedure.

Requirements

To update alternate contacts with the AWS Organizations console, you need to do some preliminary settings:

- Your organization must enable all features to manage settings on your member accounts. This allows admin control over the member accounts. This is set by default when you create your

organization. If your organization is set to consolidated billing only, and you want to enable all features, see Enabling all features in your organization.

- You need to enable trusted access for AWS Account Management service. To set this up, see Enabling trusted access for AWS Account Management.

> **Note**
>
> The AWS Organizations managed policies `AWSOrganizationsReadOnlyAccess` or `AWSOrganizationsFullAccess` are updated to provide permission to access the AWS Account Management APIs so you can access account data from the AWS Organizations console. To view the updated managed policies, see Updates to Organizations AWS managed policies.

AWS Management Console

To add or edit the alternate contacts for any AWS account in your organization

1. Sign in to the AWS Organizations console with the organization's management account credentials.
2. From **AWS accounts**, select the account that you want to update.
3. Choose **Contact info**, and under **Alternate contacts**, locate the type of contact: **Billing contact**, **Security contact**, or **Operations contact**.
4. To add a new contact, select **Add**, or to update an existing contact select **Edit**.
5. Change the values in any of the available fields.

   > **Important**
   >
   > For business AWS accounts, it's a best practice to enter a company phone number and email address rather than one belonging to an individual.

6. After you have made all of your changes, choose **Update**.

AWS CLI & SDKs

You can retrieve, update, or delete the ***alternate*** contact information by using the following AWS CLI commands or their AWS SDK equivalent operations:

- GetAlternateContact
- PutAlternateContact
- DeleteAlternateContact

Notes

- To perform these operations from the management account or a delegated admin account in an organization against member accounts, you must enable trusted access for the Account service.
- You can't access an account in a different organization from the one you're using to call the operation.

Minimum permissions

For each operation, you must have the permission that maps to that operation:

- `GetAlternateContact` (to see the alternate contact details)
- `PutAlternateContact` (to set or update an alternate contact)
- `DeleteAlternateContact` (to delete an alternate contact)

If you use these individual permissions, you can grant some users the ability to only read the contact information, and grant others the ability to both read and write.

Example

The following example retrieves the current Billing alternate contact for the caller's account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

```
$ aws account get-alternate-contact \
    --alternate-contact-type=BILLING \
    --account-id 123456789012
{
    "AlternateContact": {
        "AlternateContactType": "BILLING",
        "EmailAddress": "saanvi.sarkar@amazon.com",
        "Name": "Saanvi Sarkar",
        "PhoneNumber": "+1(206)555-0123",
        "Title": "CFO"
    }
}
```

Example

The following example sets the Operations alternate contact for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

```
$ aws account put-alternate-contact \
    --account-id 123456789012 \
    --alternate-contact-type=OPERATIONS \
    --email-address=mateo_jackson@amazon.com \
    --name="Mateo Jackson" \
    --phone-number="+1(206)555-1234" \
    --title="Operations Manager"
```

This command produces no output if it's successful.

> **Note**
>
> If you perform multiple `PutAlternateContact` operations on the same AWS account and the same contact type, the first adds the new contact, and all successive calls to the same AWS account and contact type update the existing contact.

Example

The following example deletes the Security alternate contact for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

```
$ aws account delete-alternate-contact \
    --account-id 123456789012 \
    --alternate-contact-type=SECURITY
```

This command produces no output if it's successful.

Example

> **Note**
>
> If you try to delete the same contact more than once, the first succeeds silently. All later attempts generate a `ResourceNotFound` exception.

account:AlternateContactTypes context key

You can use the context key `account:AlternateContactTypes` to specify which of the three billing types is allowed (or denied) by the IAM policy. For example, the following example IAM permission policy uses this condition key to allow the attached principals to retrieve, but not modify, only the `BILLING` alternate contact for a specific account in an organization.

Because `account:AlternateContactTypes` is a multi-valued string type, you must use the ForAnyValue or ForAllValues multi-value string operators.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "account:GetAlternateContact",
            "Resource": [
                "arn:aws:account::123456789012:account/o-aa111bb222/111111111111"
```

```
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "account:AlternateContactTypes": [
                        "BILLING"
                    ]
                }
            }
        }
    ]
}
```

Update the primary contact for your AWS account

You can update the primary contact information associated with your account, including your contact's full name, company name, mailing address, telephone number, and website address.

You edit the primary account contact differently, depending on whether or not the accounts are standalone, or part of an organization:

- **Standalone AWS accounts** – For AWS accounts not associated with an organization, you can update your own primary account contact using the AWS Management Console, or via AWS CLI & SDKs. To learn how to do this, see Update standalone AWS account primary contact.
- **AWS accounts within an organization** – For member accounts that are part of an AWS organization, a user in the management account or delegated admin account can centrally update any member account in the organization from the AWS Organizations console, or programmatically via the AWS CLI & SDKs. To learn how to do this, see Update AWS account primary contact in your organization.

Topics

Phone number and email address requirements

Before you proceed with updating your account's primary contact information, we recommend that you first review the following requirements when entering phone numbers and email addresses.

- Phone numbers can only contain numbers, whitespaces and the following characters:"+ - ( )".
- Phone numbers must start with a + and country code and must not have any leading zeros or additional spaces after the country code. For example, +1 (US/Canada) or +44 (UK).
- Phone numbers should include hyphens "-" between the area code, exchange code, and local code. For example, +1 202-555-0179.

> **Note**
>
> Phone numbers entered without hyphens may result in not being able to receive calls during the phone number verification process when resetting an MFA device for the root user. For more information, see How do I reset my AWS root user account MFA device?.

- For security purposes, phone numbers must be capable of receiving SMS from AWS. Toll free numbers will not be accepted since most don't support SMS.
- For business AWS accounts, it's a best practice to enter a company phone number and email address rather than one belonging to an individual. Configuring the accounts root user with an individual's email address or phone number can make your account difficult to recover if that individual leaves the company.

Update the primary contact for a standalone AWS account

To edit your primary contact details for a standalone AWS account, perform the steps in the following procedure. The AWS Management Console procedure below always works *only* in the standalone context. You can use the AWS Management Console to access or change only the primary contact information of the account you used to call the operation.

AWS Management Console

To edit your primary contact for a standalone AWS account

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:
>
> - `account:GetContactInformation` (to see the primary contact details)
> - `account:PutContactInformation` (to update the primary contact details)

1. Sign in to the AWS Management Console as an IAM user or role that has the minimum permissions.
2. Choose your account name on the top right of the window, and then choose **Account**.
3. Scroll down to the section **Contact information**, and next to it choose **Edit**.
4. Change the values in any of the available fields.
5. After you have made all of your changes, choose **Update**.

AWS CLI & SDKs

You can retrieve, update, or delete the ***primary*** contact information by using the following AWS CLI commands or their AWS SDK equivalent operations:

- GetContactInformation
- PutContactInformation

Notes

- To perform these operations from the management account or a delegated admin account in an organization against member accounts, you must enable trusted access for the Account service.

Minimum permissions

For each operation, you must have the permission that maps to that operation:

- `account:GetContactInformation`
- `account:PutContactInformation`

If you use these individual permissions, you can grant some users the ability to only read the contact information, and grant others the ability to both read and write.

Example

The following example retrieves the current primary contact information for the caller's account.

```
$ aws account get-contact-information
{
    "ContactInformation": {
        "AddressLine1": "123 Any Street",
        "City": "Seattle",
        "CompanyName": "Example Corp, Inc.",
        "CountryCode": "US",
        "DistrictOrCounty": "King",
        "FullName": "Saanvi Sarkar",
        "PhoneNumber": "+15555550100",
        "PostalCode": "98101",
        "StateOrRegion": "WA",
        "WebsiteUrl": "https://www.examplecorp.com"
    }
}
```

Example

The following example sets new primary contact information for the caller's account.

```
$ aws account put-contact-information --contact-information \
'{"AddressLine1": "123 Any Street", "City": "Seattle", "CompanyName": "Example Corp,
 Inc.", "CountryCode": "US", "DistrictOrCounty": "King",
"FullName": "Saanvi Sarkar", "PhoneNumber": "+15555550100", "PostalCode": "98101",
 "StateOrRegion": "WA", "WebsiteUrl": "https://www.examplecorp.com"}'
```

This command produces no output if it's successful.

Update the primary contact for any AWS account in your organization

To edit your primary contact details in any AWS account in your organization, perform the steps in the following procedure.

Additional requirements

To update primary contact with the AWS Organizations console, you need to do some preliminary settings:

- Your organization must enable all features to manage settings on your member accounts. This allows admin control over the member accounts. This is set by default when you create your organization. If your organization is set to consolidated billing only, and you want to enable all features, see Enabling all features in your organization.
- You need to enable trusted access for AWS Account Management service. To set this up, see Enabling trusted access for AWS Account Management.

AWS Management Console

To edit your primary contact for any AWS account in your organization

1. Sign in to the AWS Organizations console with the organization's management account credentials.
2. From **AWS accounts**, select the account that you want to update.
3. Choose **Contact info**, and locate **Primary contact**,
4. Select **Edit**.
5. Change the values in any of the available fields.
6. After you have made all of your changes, choose **Update**.

AWS CLI & SDKs

You can retrieve, update, or delete the ***primary*** contact information by using the following AWS CLI commands or their AWS SDK equivalent operations:

- GetContactInformation
- PutContactInformation

Notes

- To perform these operations from the management account or a delegated admin account in an organization against member accounts, you must enable trusted access for the Account service.
- You can't access an account in a different organization from the one you're using to call the operation.

Minimum permissions

For each operation, you must have the permission that maps to that operation:

- `account:GetContactInformation`
- `account:PutContactInformation`

If you use these individual permissions, you can grant some users the ability to only read the contact information, and grant others the ability to both read and write.

Example

The following example retrieves the current primary contact information for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

```
$ aws account get-contact-information --account-id 123456789012
{
    "ContactInformation": {
        "AddressLine1": "123 Any Street",
        "City": "Seattle",
        "CompanyName": "Example Corp, Inc.",
        "CountryCode": "US",
        "DistrictOrCounty": "King",
        "FullName": "Saanvi Sarkar",
        "PhoneNumber": "+15555550100",
        "PostalCode": "98101",
        "StateOrRegion": "WA",
        "WebsiteUrl": "https://www.examplecorp.com"
    }
}
```

Example

The following example sets the primary contact information for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

```
$ aws account put-contact-information --account-id 123456789012 \
--contact-information '{"AddressLine1": "123 Any Street", "City": "Seattle",
 "CompanyName": "Example Corp, Inc.", "CountryCode": "US", "DistrictOrCounty":
 "King",
"FullName": "Saanvi Sarkar", "PhoneNumber": "+15555550100", "PostalCode": "98101",
 "StateOrRegion": "WA", "WebsiteUrl": "https://www.examplecorp.com"}'
```

This command produces no output if it's successful.

Update security challenge questions

Security challenge questions are a verification method used previously to verify an identity in account recovery scenarios. They are less secure than more modern forms of verification, such as multi-factor authentication (MFA). If you currently have security challenge questions active on your AWS account, AWS Support can use these to help authenticate you as the owner of the account.

> **Important**
>
> Starting January 5, 2024, AWS will no longer support security challenge questions for accounts that have not already enabled and used them. This will remove the option to add new security challenge questions from the **Accounts** page in the AWS Management Console. If you have already set security challenge questions or have already set them on the management account in your AWS Organization, you can continue to use them. After January 6, 2025, AWS will no longer support security challenge questions for all remaining customers. We encourage you to add MFA instead. For more information, see AWS Accounts discontinues the use of security challenge questions.

To edit existing security challenge questions and provide the answers, perform the steps in the following procedure.

AWS Management Console

To edit security challenge questions for your AWS account

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:

- `account:GetChallengeQuestions` (to see the security challenge questions)
- `account:PutChallangeQuestions` (to set or update the security challenge questions)

1. Sign in to the AWS Management Console as either the AWS account root user or as an IAM user or role that has the minimum permissions.
2. Choose your account name on the top right of the window, and then choose **Account**.
3. Scroll down to the section **Security challenge questions** and choose **Edit**.

   Note

   If you don't see the **Edit** option, it is likely that you are not signed in as the root user for your account or as someone who has the minimum permissions specified above.
4. Change the values in any of the available fields. You can select any of the provided questions, and then enter the appropriate answer.
5. After you complete your changes, choose **Update**.

AWS CLI & SDKs

This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

Specify which AWS Regions your account can use

An *AWS Region* is a physical location in the world where we have multiple Availability Zones. Availability Zones consist of one or more discrete AWS data centers, each with redundant power, networking, and connectivity, housed in separate facilities. This means that each AWS Region is physically isolated and independent of the other Regions. Regions provide fault tolerance, stability, and resilience, and can also reduce latency. For a map of available and upcoming Regions, see Regions and Availability Zones.

The resources that you create in one Region do not exist in any other Region unless you explicitly use a replication feature offered by an AWS service. For example, Amazon S3 and Amazon EC2

support cross-Region replication. Some services, such as AWS Identity and Access Management (IAM), do not have Regional resources.

Your account determines the Regions that are available to you.

- An AWS account provides multiple Regions so that you can launch AWS resources in locations that meet your requirements. For example, you might want to launch Amazon EC2 instances in Europe to be closer to your European customers or to meet legal requirements.
- An AWS GovCloud (US-West) account provides access to the AWS GovCloud (US-West) Region and the AWS GovCloud (US-East) Region. For more information, see AWS GovCloud (US).
- An Amazon AWS (China) account provides access to the Beijing and Ningxia Regions only. For more information, see Amazon Web Services in China.

For a list of Region names and their corresponding codes, see Regional endpoints in the *AWS General Reference Guide*. For a list of AWS services supported in each Region (without endpoints), see the AWS Regional Services List.

> **Important**
>
> AWS recommends that you use regional AWS Security Token Service (AWS STS) endpoints instead of the global endpoint to reduce latency. Session tokens from regional AWS STS endpoints are valid in all AWS Regions. If you use regional AWS STS endpoints, you don't need to make any changes. However, session tokens from the *global* AWS STS endpoint (https://sts.amazonaws.com) are valid only in AWS Regions that you enable, or that are enabled by default. If you intend to enable a new Region for your account, you can either use session tokens from regional AWS STS endpoints or activate the global AWS STS endpoint to issue session tokens that are valid in all AWS Regions. Session tokens that are valid in all Regions are larger. If you store session tokens, these larger tokens might affect your systems. For more information about how AWS STS endpoints work with AWS Regions, see Managing AWS STS in an AWS Region.

Topics

- Considerations before enabling and disabling Regions
- Enable or disable a Region for standalone accounts
- Enable or disable a Region in your organization

Considerations before enabling and disabling Regions

Before you enable or disable a Region, it's important to consider the following:

- **Regions introduced before March 20, 2019 are enabled by default** – AWS originally enabled all new AWS Regions by default, which means you can begin creating and managing resources in these Regions immediately. You cannot enable or disable a Region that is enabled by default. Today, when AWS adds a Region, the new Region is disabled by default. If you want your users to be able to create and manage resources in a new Region, you first need to enable that Region. The following Regions are disabled by default.

Name	Code
Africa (Cape Town)	`af-south-1`
Asia Pacific (Hong Kong)	`ap-east-1`
Asia Pacific (Hyderabad)	`ap-south-2`
Asia Pacific (Jakarta)	`ap-southeast-3`
Asia Pacific (Melbourne)	`ap-southeast-4`
Canada (Calgary)	`ca-west-1`
Europe (Milan)	`eu-south-1`
Europe (Spain)	`eu-south-2`
Europe (Zurich)	`eu-central-2`
Israel (Tel Aviv)	`il-central-1`
Middle East (Bahrain)	`me-south-1`
Middle East (UAE)	`me-central-1`

- **You can use IAM permissions to control access to Regions** – AWS Identity and Access Management (IAM) includes four permissions that let you control which users can enable, disable, get, and list Regions. For more information, see Billing and Cost Management

actions policies in the *AWS Billing and Cost Management User Guide*. You can also use the `aws:RequestedRegion` condition key to control access to AWS services in an AWS Region.

- **Enabling a Region is free** – There is no charge to enable a Region. You're charged only for resources that you create in the new Region.
- **Disabling a Region disables IAM access to resources in the Region** – If you disable a Region that still contains AWS resources, such as Amazon Elastic Compute Cloud (Amazon EC2) instances, you lose IAM access to the resources in that Region. For example, you can't use the AWS Management Console to view or change the configuration of any EC2 instances in a disabled Region.
- **Charges for active resources continue if you disable a Region** – If you disable a Region that still contains AWS resources, charges for those resources (if any) continue to accrue at the standard rate. For example, if you disable a Region that contains Amazon EC2 instances, you still have to pay the charges for those instances even though the instances are inaccessible.
- **Disabling a Region isn't always immediately visible** – Services and consoles might be temporarily visible after disabling a region. Disabling a Region can takes a few minutes to several hours to take effect.
- **Enabling a Region takes a few minutes to several hours in some cases** – When you enable a Region, AWS performs actions to prepare your account in that Region, such as distributing your IAM resources to the Region. This process takes a few minutes for most accounts, but can sometimes take several hours. You cannot use the Region until this process is complete.
- **Organizations can have 50 region-opt requests open at a given time across an AWS organization** – The management account can at any point in time have 50 open requests pending completion for its organization. One request is equal to either an enable or disable of one particular region for one account.
- **A single account can have 6 region-opt requests in progress at any given time** – One request is equal to either an enable or disable of one particular region for one account.
- **Amazon EventBridge integration** – Customers can subscribe to region-opt status update notifications in EventBridge. An EventBridge notification will be created for each status change, allowing customers to automate work flows.
- **Expressive Region-opt status** – Due to the asynchronous nature of enabling/disabling an opt-in region, there are four potential statuses for a region-opt request:
  - `ENABLING`
  - `DISABLING`
  - `ENABLED`
  - `DISABLED`

You cannot cancel an opt-in or opt-out when it is in either `ENABLING` or `DISABLING` status. Otherwise, a `ConflictException` will be thrown. A completed (Enabled/Disabled) region-opt request is dependent on the provisioning of key underlying AWS services. There might be some AWS services that will not be immediately usable despite the status being `ENABLED`.

- **Full integration with AWS Organizations** – A management account can modify or read region-opt for any member account of that AWS organization. A member account is able to read/write their region state as well.

Enable or disable a Region for standalone accounts

To update which Regions your AWS account has access to, perform the steps in the following procedure. The AWS Management Console procedure below always works only in the standalone context. You can use the AWS Management Console to view or update only the available Regions in the account you used to call the operation.

AWS Management Console

To enable or disable a Region for a standalone AWS account

> **Minimum permissions**
>
> To perform the steps in the following procedure, an IAM user or role must have the following permissions:
>
> - `account:ListRegions` (needed to view the list of AWS Regions and whether they are currently enabled or disabled).
> - `account:EnableRegion`
> - `account:DisableRegion`

1. Sign in to the AWS Management Console as either the AWS account root user or as an IAM user or role that has the minimum permissions.
2. Choose your account name on the top right of the window, and then choose **Account**.
3. On the **Account** page, scroll down to the section **AWS Regions**.

> **Note**
>
> You might be prompted to approve your access to this information. AWS sends a request to the email address associated with the account and to the primary contact phone number. Choose the link in the request to open it in your browser, and approve the access.

4. Next to each AWS Region with an option in the **Action** column, choose either **Enable** or **Disable**, depending on whether you want the users in your account to be able to create and access resources in that Region.
5. If prompted, confirm your choice.
6. After you have made all of your changes, choose **Update**.

AWS CLI & SDKs

You can enable, disable, read and list region opt status by using the following AWS CLI commands or their AWS SDK equivalent operations:

- `EnableRegion`
- `DisableRegion`
- `GetRegionOptStatus`
- `ListRegions`

> **Minimum permissions**
>
> To perform the following steps, you must have the permission that maps to that operation:
>
> - `account:EnableRegion`
> - `account:DisableRegion`
> - `account:GetRegionOptStatus`
> - `account:ListRegions`

If you use these individual permissions, you can grant some users the ability to only read region opt information, and grant others the ability to both read and write.

The following example enables a region for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

Note that you can also disable a region using the same command and then replacing `enable-region` with `disable-region`.

```
aws account enable-region --region-name af-south-1
```

This command produces no output if it's successful.

The operation is asynchronous. The following command will allow you to see the latest status of the request.

```
aws account get-region-opt-status --region-name af-south-1
  {
    "RegionName": "af-south-1",
    "RegionOptStatus": "ENABLING"
  }
```

Enable or disable a Region in your organization

To update the enabled Regions for member accounts of your AWS Organizations, perform the steps in the following procedure.

> **Note**
>
> The AWS Organizations managed policies `AWSOrganizationsReadOnlyAccess` or `AWSOrganizationsFullAccess` are updated to provide permission to access the AWS Account Management APIs so you can access account data from the AWS Organizations console. To view the updated managed policies, see Updates to Organizations AWS managed policies.

Note

Before you can perform these operations from the management account or a delegated admin account in an organization for use with member accounts, you must:

- Enable all features in your organization to manage settings on your member accounts. This allows admin control over the member accounts. This is set by default when you create your organization. If your organization is set to consolidated billing only, and you want to enable all features, see Enabling all features in your organization.
- Enable trusted access for the AWS Account Management service. To set this up, see Enabling trusted access for AWS Account Management.

AWS Management Console

To enable or disable a Region in your organization

1. Sign in to the AWS Organizations console with your organization's management account credentials.
2. On the **AWS accounts** page, select the account that you want to update.
3. Choose the **Account settings** tab.
4. Under **Regions**, select the Region you want to enable or disable.
5. Choose **Actions**, and then choose either **Enable** or **Disable** option.
6. If you chose the **Enable** option, review the displayed text and then choose **Enable region**.
7. If you chose the **Disable** option, review the displayed text, type **disable** to confirm, and then choose **Disable region**.

AWS CLI & SDKs

You can enable, disable, read and list region opt status for organization member accounts by using the following AWS CLI commands or their AWS SDK equivalent operations:

- `EnableRegion`
- `DisableRegion`
- `GetRegionOptStatus`
- `ListRegions`

Minimum permissions

To perform the following steps, you must have the permission that maps to that operation:

- `account:EnableRegion`
- `account:DisableRegion`
- `account:GetRegionOptStatus`
- `account:ListRegions`

If you use these individual permissions, you can grant some users the ability to only read region opt information, and grant others the ability to both read and write.

The following example enables a region for the specified member account in an organization. The credentials used must be from either the organization's management account, or from the Account Management's delegated admin account.

Note that you can also disable a region using the same command and then replacing `enable-region` with `disable-region`.

```
aws account enable-region --account-id 123456789012 --region-name af-south-1
```

This command produces no output if it's successful.

Note

An organization can only have up to 20 region requests at a given time. Otherwise, you will receive a `TooManyRequestsException`.

The operation is asynchronous. The following command will allow you to see the latest status of the request.

```
aws account get-region-opt-status --account-id 123456789012 --region-name af-south-1
  {
    "RegionName": "af-south-1",
    "RegionOptStatus": "ENABLING"
  }
```

Create or update your AWS account alias

If you want the URL for your IAM users to contain your company name (or another easy-to-remember identifier) instead of the AWS account ID, you can create an *account alias*.

To learn how to create or update an account alias, see Creating, deleting, and listing an AWS account alias in the *IAM User Guide*.

Billing for your AWS account

For billing related procedures and tasks that are related to your AWS account, see the following topics in the *AWS Billing and Cost Management User Guide*:

- Changing which currency you use to pay your bill
- Updating and deleting tax registration numbers
- Enabling tax setting inheritance

Manage accounts in India

If you sign up for a new AWS account and choose India for your contact address, your user agreement is with Amazon Internet Services Private Limited (AISPL), a local AWS seller in India.AISPL manages your billing, and your invoice total is listed in Indian rupees (INR) instead of US dollars (USD). After you create an account with AISPL, you can't change the country in your contact information.

If you have an existing AWS account with an India address, your account is either with AWS or AISPL, depending on when you opened the account. To learn whether your account is with AWS or AISPL, see Determining which company your account is with. If you're an existing AWS customer, you can continue to use your AWS account. You also can choose to have both an AWS account and an AISPL account, although they can't be consolidated into the same AWS organization. For information about managing an AWS account, see Manage your AWS account.

If your account is with AISPL, follow the procedures in this topic to manage your account. This topic explains how to sign up for an AISPL account, edit information about your AISPL account, and add or edit your Permanent Account Number (PAN).

As part of the credit card verification during signup, AISPL charges your credit card 2 INR. AISPL refunds the 2 INR after verification is done. You might be redirected to your bank as part of the verification process.

Topics

Determine which company your account is with

AWS services are provided by both AWS and AISPL. Use this procedure to determine which seller your account is with.

AWS Management Console

To determine which company your account is with

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:
>
> - This procedure requires no special permissions.

1. Open the AWS Management Console at AWS Management Console.
2. In the page footer at the bottom of the page, look at the copyright notice. If the copyright is for Amazon Web Services, then your account is with AWS. If the copyright is for Amazon Internet Services Private Ltd., then your account is with AISPL.

AWS CLI & SDKs

This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

Create an AWS account with AISPL

AISPL is a local seller of AWS in India. Use the following procedure to sign up for an AISPL account if your contact address is in India.

AWS Management Console

To sign up for an AISPL account

> **Minimum permissions**
>
> To perform the following steps, you must have at least the following IAM permissions:
>
> - Because this operation occurs before you have an AWS account, this operation requires no AWS permissions.

1. Open the AWS Management Console, and then choose **Sign In to the Console**.
2. On the **Sign In** page, enter the email address that you want to use.
3. Under your email address, select **I am a new user**, and then choose **Sign in using our secure server**.
4. For each of the login credential fields, enter your information, and then choose **Create account**.
5. For each of the contact information fields, enter your information.
6. After you have read the customer agreement, select the terms and conditions check box, and then choose **Create Account and Continue**.
7. On the **Payment Information** page, enter the payment method that you want to use.
8. Under **PAN Information**, choose **No** if you don't have a Permanent Account Number (PAN) or want to add it later. If you have a PAN and want to add it now, choose **Yes**, and in the **PAN** field enter your PAN.
9. Choose **Verify Card and Continue**. You must provide your CVV as part of the verification process. AISPL charges your card 2 INR as part of the verification process. AISPL refunds the 2 INR after verification is done.
10. For **Provide a telephone number**, enter your phone number. If you have a phone extension, for **Ext**, enter your phone extension.
11. Choose **Call Me Now**. After a few moments, a four-digit pin will appear on your screen.

12. Accept the automated call from AISPL. On your phone keypad, enter the four-digit pin displayed on your screen.
13. Once the automated call verifies your contact number, choose **Continue to Select Your Support Plan**.
14. On the **Support Plan** page, select your support plan, and then choose **Continue**. After your payment method is verified and your account is activated, you receive an email message confirming the activation of your account.

AWS CLI & SDKs

This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

Manage your AISPL account

Except for the following tasks, the procedures for managing your account are the same as accounts created outside of India. See Manage your AWS account.

Use the AWS Management Console to perform the following tasks:

- Add or edit a Permanent Account Number (PAN)
- Edit multiple Permanent Account Numbers (PANs)
- Edit multiple Goods and Services Tax Numbers (GSTs)
- View a tax invoice

Close an AWS account

If you no longer need your AWS account, you can close it at any time by following the instructions in this section. After you've closed it, you can reopen it within 90 days from the day you closed the account. The timespan between the day you closed the account and when AWS permanently closes the account is referred to as the post-closure period.

What you need to know before closing your account

Before closing your AWS account, you should consider the following:

- Closing your account will serve as your notice of termination of the AWS Customer Agreement for this account.
- You don't need to delete resources in your AWS account before closing it. However, we recommend you back up any resources or data that you want to keep. For instructions about how to back up a particular resource, see the appropriate AWS documentation for that service.
- You can reopen your account during the post-closure period. Charges for the services that remained in your account will restart if you reopen it. You also remain responsible for any unpaid invoices and outstanding Reserved Instances and Savings Plans.
- You remain responsible for all outstanding fees and charges for the services consumed before account closure. You will receive an AWS bill the following month after closing your account. For example, if you closed your account on January 15, you will receive a bill at the beginning of February for usage incurred from January 1 through January 15. You will continue receiving invoices for Reserved Instances and Savings Plans after closing your account until they expire.
- You will no longer be able to access AWS services that were previously available in your account. However, you can sign-in and access a closed AWS account during the post-closure period only to view past billing information, access account settings, or contact AWS Support.
- You can't use the same email address that was registered to your AWS account at the time of its closure as the primary email of another AWS account. If you want to use the same email address for a different AWS account, we recommend updating it before closure. See Update the AWS account name, email address, or password for the root user for instructions on updating your email address.
- If you've enabled multi-factor authentication (MFA) on your AWS account root user, or configured an MFA device on an IAM user, MFA isn't removed automatically when you close the account. If you choose to leave MFA turned on during the 90 days post-closure period, keep the MFA device active until the post-closure period has expired in case you need to access the account during that time. Note, the hardware TOTP token devices cannot be associated with another user after the permanent closure of your account. If you would like to use the hardware TOTP token with another user later, you have the option to deactivate the hardware MFA device before closing the account. MFA devices for IAM users must be deleted by the account administrator.

Additional considerations for member accounts

- When you close a member account, that account isn't removed from the organization until after the post-closure period. During the post-closure period, a closed member account still counts

toward your quota of accounts in the organization. To avoid having the account count against the quota, see Remove a member account from your organization before closing it.

- You can only close 10% of member accounts within a rolling 30 day period. This quota is not bound by a calendar month, but starts when you close an account. Within 30 days of that initial account closure, you can't exceed the 10% account closure limit. The minimum account closure is 10 and the maximum account closure is 1000, even if 10% of accounts exceeds 1000. For more information about Organizations quotas, see Quotas for AWS Organizations.
- If you use AWS Control Tower, you need to unmanage the member account before you attempt to close the account. See Unmanage a member account in the *AWS Control Tower User Guide*.

Service specific considerations

- AWS Marketplace subscriptions aren't automatically canceled on account closure. If you have any subscriptions, first terminate all instances of your software in the subscriptions. Then, go to the **Manage subscriptions** page of the AWS Marketplace console and cancel your subscriptions.
- Domains that are registered with Route 53 are not deleted automatically. Before you close your AWS account, you have four options:
  - You can disable automatic renewal, and the domains are automatically deleted when the registration period expires. For more information, see Enabling or Disabling Automatic Renewal for a Domain in the *Amazon Route 53 Developer Guide*.
  - You can transfer the domains to another AWS account. For more information, see Transferring a Domain to a Different AWS account.
  - You can transfer the domains to another domain registrar. For more information, see Transferring a Domain from Route 53 to Another Registrar.
  - If you already closed your account, you can open a case with AWS Support for help with transferring the domain.

How to close your account

You can close your AWS account using the following procedure. Note, that there is different guidance provided in each tab depending on the type of account [standalone, member, management, and AWS GovCloud (US)] you want to close.

If you experience any issues during the process of closing your account, see Troubleshooting issues with AWS account closure.

Standalone account

A standalone account is an individually managed account that is not part of AWS Organizations.

To close a standalone account from the Accounts page

1. Sign in to the AWS Management Console as the root user in the AWS account that you want to close. You can't close an account while signed in as an IAM user or role.
2. On the navigation bar in the upper-right corner, choose your account name or number, and then choose **Account**.
3. On the **Account** page, scroll to the bottom of the page to the **Close account** section. Read and ensure that you understand the account closure process.
4. Choose the **Close account** button to initiate the account closure process.
5. Within a few minutes, you should receive an email confirmation that your account has been closed.

> **Note**
>
> This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

Member account

A member account is an AWS account that is part of AWS Organizations.

To close a member account from the AWS Organizations console

1. Sign in to the AWS Organizations console.
2. On the **AWS accounts** page, find and choose the name of the member account you want to close. You can navigate the OU hierarchy, or look at a flat list of accounts without the OU structure.
3. Choose **Close** next to the account name at the top of the page. Organizations in Consolidated billing mode won't be able to see the **Close** button in the console. To close an account in consolidated billing mode, you will need to follow the steps in the **Standalone account** tab.
4. Select each check box to acknowledge all required account closure statements.

5. Enter the member account ID and then choose **Close account**.

To close a member account from the Accounts page

Optionally, you can close an AWS member account directly from the **Accounts** page in the AWS Management Console. For step-by-step guidance, follow the instructions in the **Standalone account** tab.

To close a member account using AWS CLI and SDKs

For instructions on how to close a member account using the AWS CLI and SDKs, see Closing a member account in your organization in the *AWS Organizations User Guide*.

Management account

A management account is an AWS account that acts as the parent or root account for AWS Organizations.

> **Note**
>
> You cannot close a management account directly from the AWS Organizations console.

To close a management account from the Accounts page

1. Sign in to the AWS Management Console as the root user for the management account that you want to close. You can't close an account while signed in as an IAM user or role.
2. Verify that there are no active member accounts remaining in your organization. To do this, go to the AWS Organizations console, and make sure that all member accounts are showing Suspended next to their account names. If you have a member account that is still active, you will need to follow the account closure guidance provided in the **Member account** tab before you can move to the next step.
3. On the navigation bar in the upper-right corner, choose your account name or number, and then choose **Account**.
4. On the **Account** page, scroll to the bottom of the page to the **Close account** section. Read and ensure that you understand the account closure process.
5. Choose the **Close account** button to initiate the account closure process.
6. Within a few minutes, you should receive an email confirmation that your account has been closed.

> **Note**
>
> This task isn't supported in the AWS CLI or by an API operation from one of the AWS SDKs. You can perform this task only by using the AWS Management Console.

AWS GovCloud (US) account

An AWS GovCloud (US) account is always linked to a single standard AWS account for billing and payment purposes.

To close an AWS GovCloud (US) account

If you have an AWS account that is linked to a AWS GovCloud (US) account, you need to close the standard account before you close the AWS GovCloud (US) account. For more details, including how to back-up data and avoid unintended AWS GovCloud (US) charges, see Closing an AWS GovCloud (US) account in the *AWS GovCloud (US) User Guide*.

What to expect after you close your account

Immediately after you close your account, the following will occur:

- You will receive an email confirming the account closure to the root user's email address. If you don't receive this email within a few hours, see Troubleshooting issues with AWS account closure.
- Any member account that you close will display a SUSPENDED label next to its account name in the AWS Organizations console.
- If you have granted permissions to access services in your AWS account to other accounts, any access requests made from those accounts should fail after account closure. If you reopen your AWS account, other AWS accounts can again access your account's AWS services and resources if you granted the necessary permissions to them.

Post-closure period

The post-closure period refers to the length of time between the day you closed your account and when AWS permanently closes your AWS account. The post-closure period is 90 days. During the post-closure period, you can access your content and AWS services only by reopening your account. After the post-closure period, AWS permanently closes your AWS account, and you can no longer

reopen it. AWS will also delete any content and resources in your account. After an account has been permanently closed, its AWS account ID can never be reused.

Reopening your AWS account

Your account will permanently close in 90 days, after which you will not be able to reopen your account and AWS will delete the content remaining in your account. To reopen your account before it is permanently closed, (1) you must contact AWS Support as soon as possible, and (2) we must receive full payment of any outstanding balance, including providing required information as specified on the invoice, within 60 days from the date of account closure.

Using AWS Account Management in your organization

AWS Organizations is an AWS service that you can use to manage your AWS accounts as a group. This provides features like consolidated billing, where all of your accounts' bills are grouped together and handled by a single payer. You can also centrally manage the security of your organization by using policy based controls. For more information about AWS Organizations, see the AWS Organizations User Guide.

Trusted access

When you use AWS Organizations to manage your accounts as a group, most administrative tasks for the organization can be performed by only the organization's *management account*. By default, this includes only operations related to managing the organization itself. You can extend this additional functionality to other AWS services by enabling *trusted access* between Organizations and that service. Trusted access grants permissions to the specified AWS service to access information about the organization and the accounts it contains. When you enable trusted access for Account Management, the Account Management service grants Organizations and its management account permissions to access the metadata, such as the primary or alternate contact information, for all of the organization's member accounts.

For more information, see Enabling trusted access for AWS Account Management.

Delegated admin

After you enable trusted access, you can also choose to designate one of your member accounts as a *delegated admin* account for AWS Account Management. This allows the delegated admin account to perform the same Account Management metadata management tasks for the member accounts in your organization that previously only the management account could do. The delegated admin account can access only the management tasks for the Account Management service. The delegated admin account doesn't have all of the administrative access to the organization that the management account has.

For more information, see Enabling a delegated admin account for AWS Account Management.

Service control policies

When your AWS account is part of an organization that is managed by AWS Organizations, then the administrator of the organization can apply service control policies (SCPs) that can limit what

the principals in member accounts can do. An SCP never grants permissions; instead, it is a filter that limits what permissions can be used by the member account. A user or role (*a principal*) in a member account can perform only those operations that are in the intersection of what is allowed by the SCPs that apply to the account and the IAM permission policies attached to the principal. For example, you can use SCPs to prevent any principal in an account from modifying their own account's alternate contacts.

For example SCPs that apply to AWS accounts, see Restricting access with AWS Organizations service control policies.

Enabling trusted access for AWS Account Management

Enabling trusted access for AWS Account Management allows the administrator of the management account to modify the information and metadata (for example, primary or alternate contact details) specific to each member account in AWS Organizations. For more information, see AWS Account Management and AWS Organizations in the *AWS Organizations User Guide*. For general information about how trusted access works, see Using AWS Organizations with other AWS services.

After trusted access has been enabled, you can use the `accountID` parameter in those Account Management API operations that support it. You can use this parameter successfully only if you call the operation using credentials from the management account, or from the delegated admin account for your organization if you enable one. For more information, see Enabling a delegated admin account for AWS Account Management.

Use the following procedure to enable trusted access for Account Management in your organization.

Minimum permissions

To perform these tasks, you must meet the following requirements:

- You can perform this only from the organization's management account.
- Your organization must have all features enabled.

AWS Management Console

To enable trusted access for AWS Account Management

1. Sign in to the AWS Organizations console. You must sign in as an IAM user, assume an IAM role, or sign in as the root user (not recommended) in the organization's management account.
2. Choose **Services** in the navigation pane.
3. Choose **AWS Account Management** in the list of services.
4. Choose **Enable trusted access**.
5. In the **Enable trusted access for AWS Account Management** dialog box, type **enable** to confirm it, and then choose **Enable trusted access**.

AWS CLI & SDKs

To enable trusted access for AWS Account Management

After running the following command, you can use credentials from the organization's management account to call Account Management API operations that use the `--accountId` parameter to reference member accounts in an organization.

- AWS CLI: enable-aws-service-access

  The following example enables trusted access for AWS Account Management in the calling account's organization.

```
$ aws organizations enable-aws-service-access \
    --service-principal account.amazonaws.com
```

  This command produces no output if it's successful.

Enabling a delegated admin account for AWS Account Management

A delegated admin account can call the AWS Account Management API operations for other member accounts in the organization. To designate a member account in your organization as a delegated admin account, use the following procedure.

Minimum permissions

To perform these tasks, you must meet the following requirements:

- You can perform this only from the organization's management account.
- Your organization must have all features enabled.
- You must have enabled trusted access for Account Management in your organization.

After you specify a delegated admin account for your organization, users and roles in that account can call the AWS CLI and AWS SDK operations in the `account` namespace that can work in the Organizations mode by supporting an optional `AccountId` parameter.

AWS Management Console

This task isn't supported in the AWS Account Management management console. You can perform this task only by using the AWS CLI or an API operation from one of the AWS SDKs.

AWS CLI & SDKs

To register a delegated admin account for the Account Management service

You can use the following commands to enable a delegated admin for the Account Management service.

You must specify the following service principal:

```
account.amazonaws.com
```

- AWS CLI: register-delegated-administrator

  The following example registers a member account of the organization as a delegated admin for the Account Management service.

  ```
  $ aws organizations register-delegated-administrator \
      --account-id 123456789012 \
      --service-principal account.amazonaws.com
  ```

  This command produces no output if it's successful.

After you run this command, you can use credentials from account 123456789012 to call Account Management AWS CLI and SDK API operations that use the `--account-id` parameter to reference member accounts in an organization.

Restricting access with AWS Organizations service control policies

This topic presents examples that show how you can use service control policies (SCPs) to restrict what the users and roles in the accounts in your organization can do. For more information about service control policies, see the following topics in the *AWS Organizations User Guide*:

- Creating SCPs
- Attaching SCPs to OUs and accounts
- Strategies for SCPs
- SCP policy syntax

Example Example 1: Prevent accounts from modifying their own alternate contacts

The following example denies the `PutAlternateContact` and `DeleteAlternateContact` API operations from being called by any member account in standalone account mode. This prevents any principal in the affected accounts from changing their own alternate contacts.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Deny",
            "Action": [
                "account:PutAlternateContact",
                "account:DeleteAlternateContact"
            ],
            "Resource": [ "arn:aws:account::*:account" ]
        }
    ]
}
```

Example Example 2: Prevent any member account from modifying alternate contacts for any other member account in the organization

The following example generalizes the `Resource` element to "*", which means that it applies to both standalone mode requests and organizations mode requests. This means that even the delegated admin account for Account Management, if the SCP applies to it, is blocked from changing any alternate contact for any account in the organization.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Deny",
            "Action": [
                "account:PutAlternateContact",
                "account:DeleteAlternateContact"
            ],
            "Resource": [ "*" ]
        }
    ]
}
```

Example Example 3: Prevent a member account in an OU from modifying its own alternate contacts

The following example SCP includes a condition that compares the account's organization path to a list of two OUs. This results in blocking a principal in any account in the specified OUs from modifying their own alternate contacts.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Deny",
            "Action": "account:PutAlternateContact",
            "Resource": [
                "arn:aws:account::*:account"
            ],
            "Condition": {
                "ForAnyValue:StringLike": {
```

```
                    "account:AccountResourceOrgPath": [
                        "o-aa111bb222/r-a1b2/ou-a1b2-f6g7h111/",
                        "o-aa111bb222/r-a1b2/ou-a1b2-f6g7h222/"
                    ]
                }
            }
    ]
}
```

Security in AWS Account Management

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 Account Management, 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 Account Management. It shows you how to configure Account Management to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Account Management resources.

Topics

Data protection in AWS Account Management

The AWS shared responsibility model applies to data protection in AWS Account Management. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

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

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

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

AWS PrivateLink for AWS Account Management

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can access the AWS Account Management service from within the VPC without having to cross the public internet.

Amazon VPC lets you launch AWS resources in a custom virtual network. You can use a VPC to control your network settings, such as the IP address range, subnets, route tables, and network gateways. For more information about VPCs, see the *Amazon VPC User Guide*.

To connect your Amazon VPC to Account Management, you must first define an *interface VPC endpoint*, which lets you connect your VPC to other AWS services. The endpoint provides reliable, scalable connectivity, without requiring an internet gateway, network address translation (NAT) instance, or VPN connection. For more information, see Interface VPC Endpoints (AWS PrivateLink) in the *Amazon VPC User Guide*.

Creating the Endpoint

You can create an AWS Account Management endpoint in your VPC using the AWS Management Console, the AWS Command Line Interface (AWS CLI), an AWS SDK, the AWS Account Management API, or AWS CloudFormation.

For information about creating and configuring an endpoint using the Amazon VPC console or the AWS CLI, see Creating an Interface Endpoint in the *Amazon VPC User Guide.*

Note

When you create an endpoint, specify Account Management as the service that you want your VPC to connect to, using the following format:

```
com.amazonaws.us-east-1.account
```

You must use the string exactly as shown, specifying the `us-east-1` Region. As a global service, Account Management is hosted in only that one AWS Region.

For information about creating and configuring an endpoint using AWS CloudFormation, see the AWS::EC2::VPCEndpoint resource in the *AWS CloudFormation User Guide*.

Amazon VPC Endpoint Policies

You can control what actions can performed through this service endpoint by attaching an endpoint policy when you create the Amazon VPC endpoint. You can create complex IAM rules by attaching multiple endpoint policies. For more information, see:

- Amazon Virtual Private Cloud endpoint policies for Account Management
- Controlling Access to Services with VPC Endpoints in the *AWS PrivateLink Guide*.

Amazon Virtual Private Cloud endpoint policies for Account Management

You can create a Amazon VPC endpoint policy for Account Management in which you specify the following:

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

The following example shows an Amazon VPC endpoint policy that allows one IAM user named Alice in account 123456789012 to both retrieve and change the alternate contact information for any AWS account, but denies all IAM users permission to delete any alternate contact information on any account.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "account:GetAlternateContact",
                "account:PutAlternateContact"
            ],
            "Resource": "arn:aws::iam:*:account,
            "Effect": "Allow",
            "Principal": {
              "AWS": "arn:aws::iam:123456789012:user/Alice"
            }
        },
```

```
        {
            "Action": "account:DeleteAlternateContact",
            "Resource": "*",
            "Effect": "Deny",
            "Principal": "arn:aws::iam:*:root"
        }
    ]
}
```

If you want to grant access to accounts that are part of an AWS Organization to a principal that is in one of the organization's member accounts, then the `Resource` element must use the following format:

```
arn:aws:account::{ManagementAccountId}:account/o-{OrganizationId}/{AccountId}
```

For more information about creating endpoint policies, see Controlling Access to Services with VPC Endpoints in the *AWS PrivateLink Guide*.

Identity and Access Management for AWS Account Management

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 Account Management resources. IAM is an AWS service that you can use with no additional charge.

Topics

- Audience
- Authenticating with identities
- Managing access using policies
- How AWS Account Management works with IAM
- Identity-based policy examples for AWS Account Management
- Using identity-based policies (IAM policies) for AWS Account Management
- Troubleshooting AWS Account Management identity and access

Audience

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

Service user – If you use the Account Management service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Account Management 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 Account Management, see Troubleshooting AWS Account Management identity and access.

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

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Account Management. To view example Account Management identity-based policies that you can use in IAM, see Identity-based policy examples for AWS Account Management.

Authenticating with identities

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

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

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

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signing AWS API requests in the *IAM User Guide*.

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

AWS account root user

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

Federated identity

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

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

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

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

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

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

IAM roles

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

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

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an 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 AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

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

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

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*.

Resource-based policies

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

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

Access control lists (ACLs)

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

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

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

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If

you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

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

How AWS Account Management works with IAM

Before you use IAM to manage access to Account Management, learn what IAM features are available to use with Account Management.

IAM features you can use with AWS Account Management

IAM feature	Account Management support
Identity-based policies	Yes
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys	Yes
ACLs	No
ABAC (tags in policies)	Yes

IAM feature	Account Management support
Temporary credentials	Yes
Principal permissions	Yes
Service roles	No
Service-linked roles	No

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

Identity-based policies for Account Management

Supports identity-based policies	Yes

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

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for Account Management

To view examples of Account Management identity-based policies, see Identity-based policy examples for AWS Account Management.

Resource-based policies within Account Management

Supports resource-based policies	No

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

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Policy actions for Account Management

Supports policy actions	Yes

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

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. 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.

To see a list of Account Management actions, see Actions defined by AWS Account Management in the *Service Authorization Reference*.

Policy actions in Account Management use the following prefix before the action.

```
account
```

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

```
"Action": [
      "account:action1",
      "account:action2"
         ]
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that work with an AWS account's alternate contacts, include the following action.

```
"Action": "account:*AlternateContact"
```

To view examples of Account Management identity-based policies, see Identity-based policy examples for AWS Account Management.

Policy resources for Account Management

Supports policy resources	Yes

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

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

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

The Account Management service supports the following specific resource types in an IAM policy's `Resources` element to help you filter the policy and distinguish between these types of AWS accounts:

- **account**

This `resource` type matches only standalone AWS accounts that are not member accounts in an organization managed by the AWS Organizations service.

- **accountInOrganization**

  This `resource` type matches only AWS accounts that are member accounts in an organization managed by the AWS Organizations service.

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

To view examples of Account Management identity-based policies, see Identity-based policy examples for AWS Account Management.

Policy condition keys for Account Management

Supports service-specific policy condition keys	Yes

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

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

The Account Management service supports the following condition keys that you can use to provide fine-grained filtering for your IAM policies:

- **account:TargetRegion**

  This condition key takes an argument that consists of a list of AWS Region codes. It lets you filter the policy to affect only those actions that apply to the specified Regions.

- **account:AlternateContactTypes**

  This condition key takes a list of alternate contact types:

  - BILLING
  - OPERATIONS
  - SECURITY

  Using this key lets you filter the request to only those actions that target the specified alternate contact types.

- **account:AccountResourceOrgPaths**

  This condition key takes an argument that consists of a list of ARNs with wildcards that represent accounts in an organization. It lets you filter the policy to affect only those actions that target accounts with ARNs that match. For example, the following ARN matches only those accounts in the specified organization and the specified organizational unit (OU).

  ```
  arn:aws:account::111111111111:ou/o-aa111bb222/r-a1b2/ou-a1b2-f6g7h111/*
  ```

- **account:AccountResourceOrgTags**

  This condition key takes an argument that consists of a list of tag keys and values. It lets you filter the policy to affect only those accounts that are members of an organization and that are tagged with the specified tag keys and values.

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

To view examples of Account Management identity-based policies, see Identity-based policy examples for AWS Account Management.

Access control lists in Account Management

Supports ACLs	No

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

Attribute-based access control with Account Management

Supports ABAC (tags in policies)	Yes

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

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

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

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

Using temporary credentials with Account Management

Supports temporary credentials	Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see Switching to a role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Cross-service principal permissions for Account Management

Supports forward access sessions (FAS)	Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

Service roles for Account Management

Supports service roles	No

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 roles for Account Management

Supports service-linked roles	No

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

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

Identity-based policy examples for AWS Account Management

By default, users and roles don't have permission to create or modify Account Management resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

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

For details about actions and resource types defined by Account Management, including the format of the ARNs for each of the resource types, see Actions, resources, and condition keys for AWS Account Management in the *Service Authorization Reference*.

Topics

- Policy best practices
- Using the Account page in the AWS Management Console
- Providing read-only access to the Account page in the AWS Management Console
- Providing full access to the Account page in the AWS Management Console

Policy best practices

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

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see IAM Access Analyzer policy validation in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Configuring MFA-protected API access in the *IAM User Guide*.

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

Using the Account page in the AWS Management Console

To access the **Account** page in the AWS Management Console, you must have a minimum set of permissions. These permissions must allow you to list and view details about 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.

To ensure that users and roles can use the Account Management console, you can choose to attach either the `AWSAccountManagementReadOnlyAccess` or `AWSAccountManagementFullAccess` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, in many cases you can choose to allow access to only the actions that match the API operations that you're trying to perform.

Providing read-only access to the Account page in the AWS Management Console

In the following example, you want to grant an IAM user in your AWS account read-only access to the Account page in the AWS Management Console. Users with this policy attached can't make any changes.

The `account:GetAccountInformation` action grants access to view most of the settings on the Account page. However, to view the currently enabled AWS Regions, you must also include the `account:ListRegions` action.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GrantReadOnlyAccessToAccountSettings",
            "Effect": "Allow",
            "Action": [
                "account:GetAccountInformation",
                "account:ListRegions"
            ],
            "Resource": "*"
        }
    ]
}
```

Providing full access to the Account page in the AWS Management Console

In the following example, you want to grant an IAM user in your AWS account full access to the Account page in the AWS Management Console. Users with this policy attached can alter settings for the account.

This example policy builds on the preceding example policy by adding each of the available write permissions (with the exception of CloseAccount), which allows the user to change most of the settings for the account, including the `account:EnableRegion` and `account:DisableRegion` permissions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GrantFullAccessToAccountSettings",
            "Effect": "Allow",
            "Action": [
                "account:GetAccountInformation",
                "account:ListRegions",
                "account:PutContactInformation",
                "account:PutChallengeQuestions",
                "account:PutAlternateContact",
                "account:DeleteAlternateContact",
                "account:EnableRegion",
                "account:DisableRegion"
            ],
            "Resource": "*"
        }
    ]
}
```

Using identity-based policies (IAM policies) for AWS Account Management

For a full discussion of AWS accounts and IAM users, see What Is IAM? in the *IAM User Guide*.

For instructions on how you can update customer managed policies, see Editing customer managed policies (console) in the *IAM User Guide*.

AWS Account Management actions policies

This table summarizes the permissions that grant access to your account settings. For examples of policies that use these permissions, see AWS Account Management policy examples.

> **Note**
>
> To grant IAM users write access to a specific account setting in the **Account** page of the AWS Management Console, you must allow the `GetAccountInformation` permission, in addition to the permission (or permissions) that you want to use to modify that setting.

Permission name	Access level	Description
`account:ListRegions`	List	Grants permission to list the available Regions.
`account:GetAccountInformation`	Read	Grants permission to retrieve the account information for an account.
`account:GetAlternateContact`	Read	Grants permission to retrieve the alternate contacts for an account.
`account:GetChallengeQuestions`	Read	Grants permission to retrieve the challenge questions for an account.
`account:GetContactInformation`	Read	Grants permission to retrieve the primary contact information for an account.
`account:GetRegionOptStatus`	Read	Grants permission to get the opt-in status of a Region.
`account:CloseAccount`	Write	Grants permission to close an account.

Permission name	Access level	Description
		Note This is a permission for the console only. No API access is available for this permission.
`account:DeleteAlternateContact`	Write	Grants permission to delete the alternate contacts for an account.
`account:DisableRegion`	Write	Grants permission to disable use of a Region.
`account:EnableRegion`	Write	Grants permission to enable use of a Region.
`account:PutAlternateContact`	Write	Grants permission to modify the alternate contacts for an account.
`account:PutChallengeQuestions`	Write	Grants permission to modify the challenge questions for an account. **Note** This is a permission for the console only. No API access is available for this permission.

Permission name	Access level	Description
account:PutContact Information	Write	Grants permission to update the primary contact informati on for an account.

Troubleshooting AWS Account Management identity and access

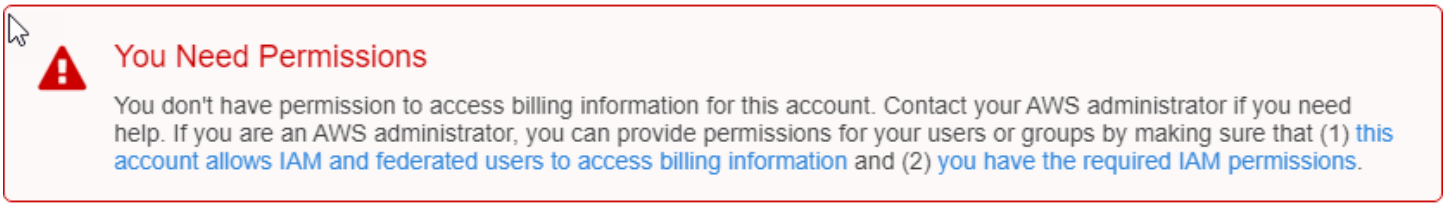
Use the following information to help you diagnose and fix common issues that you might encounter when working with Account Management and IAM.

Topics

I am not authorized to perform an action in the Account page

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

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about his AWS account in the **Account** page of the AWS Management Console but doesn't have the `account:GetAccountInformation` permissions.



In this case, Mateo asks his administrator to update his policies to allow him to access the *my-example-widget* resource using the `account:GetWidget` action.

I am not authorized to perform `iam:PassRole`

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

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

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Account Management. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

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

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

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

I want to allow people outside of my AWS account to access my account details

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

AWS managed policies for AWS Account Management

AWS Account Management currently provides two AWS managed policies that are available for your use:

- AWS managed policy: AWSAccountManagementReadOnlyAccess
- AWS managed policy: AWSAccountManagementFullAccess
- Account Management updates to AWS managed policies

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases so that you can start assigning permissions to users, groups, and roles.

Keep in mind that AWS managed policies might not grant least-privilege permissions for your specific use cases because they're available for all AWS customers to use. We recommend that you reduce permissions further by defining customer managed policies that are specific to your use cases.

You cannot change the permissions defined in AWS managed policies. If AWS updates the permissions defined in an AWS managed policy, the update affects all principal identities (users, groups, and roles) that the policy is attached to. AWS is most likely to update an AWS managed policy when a new AWS service is launched or new API operations become available for existing services.

For more information, see AWS managed policies in the *IAM User Guide*.

AWS managed policy: AWSAccountManagementReadOnlyAccess

You can attach the `AWSAccountManagementReadOnlyAccess` policy to your IAM identities.

This policy provides read-only permissions to only view the following:

- The metadata about your AWS accounts
- The AWS Regions that are enabled or disabled for the AWS account (you can view status of Regions in your account only by using the AWS console)

It does this by granting permission to run any of the `Get*` or `List*` operations. It doesn't provide any ability to modify the account metadata or enable or disable AWS Regions for the account.

Permissions details

This policy includes the following permissions.

- `account` – Allows principals to retrieve the metadata information about AWS accounts. It also allows principals to list the AWS Regions that are enabled for the account in the AWS Management Console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "account:Get*",
                "account:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS managed policy: AWSAccountManagementFullAccess

You can attach the `AWSAccountManagementFullAccess` policy to your IAM identities.

This policy provides full administrative access to view or modify the following:

- The metadata about your AWS accounts
- The AWS Regions that are enabled or disabled for the AWS account (you can view status or enable or disable Regions for your account only by using the AWS console)

It does this by granting permission to run any `account` operations.

Permissions details

This policy includes the following permissions.

- `account` – Allows principals to view or modify the metadata information about AWS accounts. It also allows principals to list the AWS Regions that are enabled for the account and enable or disable them in the AWS Management Console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "account:*",
            "Resource": "*"
        }
    ]
}
```

Account Management updates to AWS managed policies

View details about updates to AWS managed policies for Account Management since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Account Management Document history page.

Change	Description	Date
AWS Account Management launched with new AWS managed policies and started tracking changes	Account Management initially launched with the following AWS managed policies: • AWSAccountManagementReadOnlyAccess • AWSAccountManagementFullAccess	Sept 30, 2021

Compliance validation for AWS Account Management

Third-party auditors assess the security and compliance of AWS services that can run in your AWS account as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

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 in the *AWS Artifact User Guide*.

Your compliance responsibility when using services in your AWS account 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 baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance on Amazon Web Services – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.

  > **Note**
  >
  > Not all AWS services are HIPAA eligible. For more information, see the HIPAA Eligible Services Reference.

- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config 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.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in AWS Account Management

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones

without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in AWS Account Management

As managed services, AWS services running in your AWS account are protected by the AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well-Architected Framework*.

You use AWS published API calls to access account settings through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, 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.

Monitoring AWS Account Management

Monitoring is an important part of maintaining the reliability, availability, and performance of AWS Account Management and your other AWS solutions. AWS provides the following monitoring tools to watch Account Management, report when something is wrong, and take automatic actions when appropriate:

- *AWS CloudTrail* captures (logs) API calls and related events made by or on behalf of your AWS account and writes the log files to an Amazon Simple Storage Service (Amazon S3) bucket that you specify. This lets you identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. For more information, see the AWS CloudTrail User Guide.
- *Amazon EventBridge* adds additional automation to your AWS services by responding automatically to system events, such as application availability issues or resource changes. Events from AWS services are delivered to EventBridge in near real time. You can write simple rules to indicate which events are of interest to you and which automated actions to take when an event matches a rule. For more information, see the Amazon EventBridge User Guide.

Logging AWS Account Management API calls using AWS CloudTrail

The AWS Account Management APIs are integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service that calls an Account Management operation. CloudTrail captures all Account Management API calls as events. The calls captured include all calls to the Account Management operations. If you create a trail, you can turn on continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Account Management operations. 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 called an Account Management operation, the IP address used to make the request, who made the request and when, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

Account Management information in CloudTrail

CloudTrail is turned on in your AWS account when you create the account. When activity occurs with an Account Management operation, CloudTrail records that activity 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 Account Management operations, 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 AWS Management 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. You can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following:

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

AWS CloudTrail logs all Account Management API operations found in the API Reference section of this guide. For example, calls to the `CreateAccount`, `DeleteAlternateContact`, and `PutAlternateContact` operations 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 user or AWS Identity and Access Management (IAM) user credentials
- Whether the request was made with temporary security credentials for an IAM role or federated user
- Whether the request was made by another AWS service

For more information, see the CloudTrail userIdentity element.

Understanding the Account Management log 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 operation, the date and time of the operation, 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.

Example 1: The following example shows a CloudTrail log entry for a call to the `GetAlternateContact` operation to retrieve the current `OPERATIONS` alternate contact for an account. The values returned by the operation aren't included in the logged information.

Example Example 1


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA1234567890EXAMPLE:AccountAPITests",
    "arn":"arn:aws:sts::123456789012:assumed-role/ServiceTestRole/AccountAPITests",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROA1234567890EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/ServiceTestRole",
        "accountId": "123456789012",
        "userName": "ServiceTestRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-04-30T19:25:53Z"
      }
    }
  },
  "eventTime": "2021-04-30T19:26:15Z",
  "eventSource": "account.amazonaws.com",
  "eventName": "GetAlternateContact",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "10.24.34.250",
```

```
  "userAgent": "Mozilla/5.0",
  "requestParameters": {
    "alternateContactType": "SECURITY"
  },
  "responseElements": null,
  "requestID": "1a2b3c4d-5e6f-1234-abcd-111111111111",
  "eventID": "1a2b3c4d-5e6f-1234-abcd-222222222222",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012"
}
```

Example 2: The following example shows a CloudTrail log entry for a call to the `PutAlternateContact` operation to add a new `BILLING` alternate contact to an account.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA1234567890EXAMPLE:AccountAPITests",
    "arn": "arn:aws:sts::123456789012:assumed-role/ServiceTestRole/AccountAPITests",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROA1234567890EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/ServiceTestRole",
        "accountId": "123456789012",
        "userName": "ServiceTestRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-04-30T18:33:00Z"
      }
    }
  },
  "eventTime": "2021-04-30T18:33:08Z",
  "eventSource": "account.amazonaws.com",
  "eventName": "PutAlternateContact",
```

```
  "awsRegion": "us-east-1",
  "sourceIPAddress": "10.24.34.250",
  "userAgent": "Mozilla/5.0",
  "requestParameters": {
    "name": "*Alejandro Rosalez*",
    "emailAddress": "alrosalez@example.com",
    "title": "CFO",
    "alternateContactType": "BILLING"
  },
  "responseElements": null,
  "requestID": "1a2b3c4d-5e6f-1234-abcd-333333333333",
  "eventID": "1a2b3c4d-5e6f-1234-abcd-444444444444",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012"
}
```

Example 3: The following example shows a CloudTrail log entry for a call to the `DeleteAlternateContact` operation to delete the current `OPERATIONS` alternate contact.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA1234567890EXAMPLE:AccountAPITests",
    "arn":"arn:aws:sts::123456789012:assumed-role/ServiceTestRole/AccountAPITests",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROA1234567890EXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/ServiceTestRole",
        "accountId": "123456789012",
        "userName": "ServiceTestRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-04-30T18:33:00Z"
      }
```

```
    }
  },
  "eventTime": "2021-04-30T18:33:16Z",
  "eventSource": "account.amazonaws.com",
  "eventName": "DeleteAlternateContact",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "10.24.34.250",
  "userAgent": "Mozilla/5.0",
  "requestParameters": {
    "alternateContactType": "OPERATIONS"
  },
  "responseElements": null,
  "requestID": "1a2b3c4d-5e6f-1234-abcd-555555555555",
  "eventID": "1a2b3c4d-5e6f-1234-abcd-666666666666",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012"
}
```


Monitoring Account Management events with EventBridge

Amazon EventBridge, formerly called CloudWatch Events, helps you monitor events that are specific to and initiate target actions that use other AWS services. Events from AWS services are delivered to EventBridge in near real time.

Using EventBridge, you can create *rules* that match incoming *events* and route them to *targets* for processing.

For more information, see Getting started with Amazon EventBridge in the *Amazon EventBridge User Guide*.

Account Management events

The following examples show events for Account Management. Events are produced on a best-effort basis.

Only events that are specific to enabling and disabling Regions and API calls via CloudTrail are currently available for Account Management.

Event types

- Event for enabling and disabling Regions

Event for enabling and disabling Regions

When you enable or disable a Region in an account, either from the Console or from the API, an asynchronous task is kicked off. The initial request will be logged as a CloudTrail event in the target account. In addition, an EventBridge event will be sent to the calling account when either the enable or disable process has started, and again once either process has completed.

The following example event shows how a request will be sent indicating that on `2020-09-30` the `ap-east-1` Region was `ENABLED` for account `123456789012`.

```
{
   "version":"0",
   "id":"11112222-3333-4444-5555-666677778888",
   "detail-type":"Region Opt-In Status Change",
   "source":"aws.account",
   "account":"123456789012",
   "time":"2020-09-30T06:51:08Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:account::123456789012:account"
   ],
   "detail":{
      "accountId":"123456789012",
      "regionName":"ap-east-1",
      "status":"ENABLED"
   }
}
```

There are four possible statuses which match the statuses returned by the `GetRegionOptStatus` and `ListRegions` APIs:

- `ENABLED` – The Region has been successfully enabled for the `accountId` indicated
- `ENABLING` – The Region is in the process of being enabled for the `accountId` indicated
- `DISABLED` – The Region has been successfully disabled for the `accountId` indicated
- `DISABLING` – The Region is in the process of being disabled for the `accountId` indicated

The following sample event pattern, creates a rule that captures all Region events.

```
{
   "source":[
      "aws.account"
   ],
   "detail-type":[
      "Region Opt-In Status Change"
   ]
}
```

The following sample event pattern, creates a rule that captures only `ENABLED` and `DISABLED` Region events.

```
{
   "source":[
      "aws.account"
   ],
   "detail-type":[
      "Region Opt-In Status Change"
   ],
   "detail":{
      "status":[
         "DISABLED",
         "ENABLED"
      ]
   }
}
```

API Reference

The API operations in the Account Management (`account`) namespace enable you to modify your AWS account.

Every AWS account supports metadata with information about the account, including information about up to three alternate contacts associated with the account. These are in addition to the email address associated with the root user of the account. You can specify only one of each of the following contact types associated with an account.

- Billing contact
- Operations contact
- Security contact

By default, the API operations discussed in this guide apply directly to the account that calls the operation. The identity in the account that is calling the operation is typically an IAM role or IAM user and must have permission applied by an IAM policy to call the API operation. Alternatively, you can call these API operations from an identity in an AWS Organizations management account and specify the account ID number for any AWS account that is a member of the organization.

API version

This version of the Accounts API Reference documents the Account Management API version 2021-02-01.

> **Note**
>
> As an alternative to using the API directly, you can use one of the AWS SDKs, which consist of libraries and sample code for various programming languages and platforms (Java, Ruby, .NET, iOS, Android, and more). The SDKs provide a convenient way to create programmatic access to AWS Organizations. For example, the SDKs take care of cryptographically signing requests, managing errors, and retrying requests automatically. For more information about the AWS SDKs, including how to download and install them, see Tools for Amazon Web Services.

We recommend that you use the AWS SDKs to make programmatic API calls to the Account Management service. However, you also can use the Account Management Query API to make direct calls to the Account Management web service. To learn more about the Account Management Query API, see Calling the API by making HTTP Query requests in the Account Management User Guide. Organizations supports GET and POST requests for all actions. That is, the API does not require you to use GET for some actions and POST for others. However, GET requests are subject to the limitation size of a URL. Therefore, for operations that require larger sizes, use a POST request.

Signing requests

When you send HTTP requests to AWS, you must sign the requests so that AWS can identify who sent them. You sign requests with your AWS access key, which consists of an access key ID and a secret access key. We strongly recommend that you do not create an access key for your root account. Anyone who has the access key for your root account has unrestricted access to all the resources in your account. Instead, create an access key for an IAM user that has administrative privileges. As another option, use AWS Security Token Service to generate temporary security credentials, and use those credentials to sign requests.

To sign requests, we recommend that you use Signature Version 4. If you have an existing application that uses Signature Version 2, you do not have to update it to use Signature Version 4. However, some operations now require Signature Version 4. The documentation for operations that require version 4 indicate this requirement. For more information, see Signing AWS API requests in the *IAM User Guide*.

When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools.

Support and feedback for Account Management

We welcome your feedback. Send your comments to feedback-awsaccounts@amazon.com or post your feedback and questions in the Account Management support forum. For more information about the AWS support forums, see Forums Help.

How examples are presented

The JSON returned by the Account Management as response to your requests is returned as a single long string without line breaks or formatting whitespace. Both line breaks and whitespace are shown in the examples in this guide to improve readability. When example input parameters

also would result in long strings that would extend beyond the screen, we insert line breaks to enhance readability. You should always submit the input as a single JSON text string.

Recording API Requests

Account Management supports CloudTrail, a service that records AWS API calls for your AWS account and delivers log files to an Amazon S3 bucket. By using information collected by CloudTrail, you can determine which requests were successfully made to Account Management, who made the request, when it was made, and so on. For more about Account Management and its support for CloudTrail, see Logging AWS Account Management API calls using AWS CloudTrail. To learn more about CloudTrail, including how to turn it on and find your log files, see the AWS CloudTrail User Guide.

Actions

The following actions are supported:

- DeleteAlternateContact
- DisableRegion
- EnableRegion
- GetAlternateContact
- GetContactInformation
- GetRegionOptStatus
- ListRegions
- PutAlternateContact
- PutContactInformation

DeleteAlternateContact

Deletes the specified alternate contact from an AWS account.

For complete details about how to use the alternate contact operations, see Access or updating the alternate contacts.

> **Note**
>
> Before you can update the alternate contact information for an AWS account that is managed by AWS Organizations, you must first enable integration between AWS Account Management and Organizations. For more information, see Enabling trusted access for AWS Account Management.

Request Syntax

```
POST /deleteAlternateContact HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "AlternateContactType": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12 digit account ID number of the AWS account that you want to access or modify with this operation.

If you do not specify this parameter, it defaults to the AWS account of the identity used to call the operation.

To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account, and the specified account ID must be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`; it must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, then don't specify this parameter, and call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

AlternateContactType

Specifies which of the alternate contacts to delete.

Type: String

Valid Values: `BILLING | OPERATIONS | SECURITY`

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

ResourceNotFoundException

The operation failed because it specified a resource that can't be found.

HTTP Status Code: 404

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

Examples

Example 1

The following example deletes the security alternate contact for the account whose credentials are used to call the operation.

Sample Request

```
POST / HTTP/1.1
```

```
X-Amz-Target: AWSAccountV20210201.DeleteAlternateContact

{ "AlternateContactType": "SECURITY" }
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json
```

Example 2

The following example deletes the billing alternate contact for the specified member account in an organization. You must use credentials from either the organization's management account or from the Account Management service's delegated admin account.

Sample Request

```
POST / HTTP/1.1
X-Amz-Target: AWSAccountV20210201.DeleteAlternateContact

{ "AccountId": "123456789012", "AlternateContactType": "BILLING" }
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3

- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

DisableRegion

Disables (opts-out) a particular Region for an account.

> **Note**
>
> The act of disabling a Region will remove all IAM access to any resources that reside in that Region.

Request Syntax

```
POST /disableRegion HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "RegionName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

RegionName

Specifies the Region-code for a given Region name (for example, `af-south-1`). When you disable a Region, AWS performs actions to deactivate that Region in your account, such as destroying IAM resources in the Region. This process takes a few minutes for most accounts, but this can take several hours. You cannot enable the Region until the disabling process is fully completed.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

ConflictException

The request could not be processed because of a conflict in the current status of the resource. For example, this happens if you try to enable a Region that is currently being disabled (in a status of DISABLING).

HTTP Status Code: 409

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2

- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

EnableRegion

Enables (opts-in) a particular Region for an account.

Request Syntax

```
POST /enableRegion HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "RegionName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: ^\d{12}$

Required: No

RegionName

Specifies the Region-code for a given Region name (for example, af-south-1). When you enable a Region, AWS performs actions to prepare your account in that Region, such as distributing your IAM resources to the Region. This process takes a few minutes for most accounts, but it can take several hours. You cannot use the Region until this process is complete. Furthermore, you cannot disable the Region until the enabling process is fully completed.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

ConflictException

The request could not be processed because of a conflict in the current status of the resource. For example, this happens if you try to enable a Region that is currently being disabled (in a status of DISABLING).

HTTP Status Code: 409

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetAlternateContact

Retrieves the specified alternate contact attached to an AWS account.

For complete details about how to use the alternate contact operations, see Access or updating the alternate contacts.

> **Note**
>
> Before you can update the alternate contact information for an AWS account that is managed by AWS Organizations, you must first enable integration between AWS Account Management and Organizations. For more information, see Enabling trusted access for AWS Account Management.

Request Syntax

```
POST /getAlternateContact HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "AlternateContactType": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12 digit account ID number of the AWS account that you want to access or modify with this operation.

If you do not specify this parameter, it defaults to the AWS account of the identity used to call the operation.

To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account, and the specified account ID must be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`; it must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, then don't specify this parameter, and call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

AlternateContactType

Specifies which alternate contact you want to retrieve.

Type: String

Valid Values: `BILLING | OPERATIONS | SECURITY`

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "AlternateContact": {
      "AlternateContactType": "string",
      "EmailAddress": "string",
      "Name": "string",
```

```
        "PhoneNumber": "string",
        "Title": "string"
    }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

AlternateContact

A structure that contains the details for the specified alternate contact.

Type: AlternateContact object

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

ResourceNotFoundException

The operation failed because it specified a resource that can't be found.

HTTP Status Code: 404

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

Examples

Example 1

The following example retrieves the security alternate contact for the account whose credentials are used to call the operation.

Sample Request

```
POST / HTTP/1.1
X-Amz-Target: AWSAccountV20210201.GetAlternateContact

{ "AlternateContactType": "SECURITY" }
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json{
   "AlternateContact": {
        "Name": "Anika",
        "Title": "COO",
        "EmailAddress": "anika@example.com",
        "PhoneNumber": "206-555-0198"
        "AlternateContactType": "Security"
    }
}
```

Example 2

The following example retrieves the operations alternate contact for the specified member account in an organization. You must use credentials from either the organization's management account or from the Account Management service's delegated admin account.

Sample Request

```
POST / HTTP/1.1
```

```
X-Amz-Target: AWSAccountV20210201.GetAlternateContact

{ "AccountId": "123456789012", "AlternateContactType": "Operations" }
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json{
   "AlternateContact": {
        "Name": "Anika",
        "Title": "COO",
        "EmailAddress": "anika@example.com",
        "PhoneNumber": "206-555-0198"
        "AlternateContactType": "Operations"
    }
}
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetContactInformation

Retrieves the primary contact information of an AWS account.

For complete details about how to use the primary contact operations, see Update the primary and alternate contact information.

Request Syntax

```
POST /getContactInformation HTTP/1.1
Content-type: application/json

{
   "AccountId": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: ^\d{12}$

Required: No

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "ContactInformation": {
      "AddressLine1": "string",
      "AddressLine2": "string",
      "AddressLine3": "string",
      "City": "string",
      "CompanyName": "string",
      "CountryCode": "string",
      "DistrictOrCounty": "string",
      "FullName": "string",
      "PhoneNumber": "string",
      "PostalCode": "string",
      "StateOrRegion": "string",
      "WebsiteUrl": "string"
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

ContactInformation

Contains the details of the primary contact information associated with an AWS account.

Type: ContactInformation object

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

ResourceNotFoundException

The operation failed because it specified a resource that can't be found.

HTTP Status Code: 404

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET

- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetRegionOptStatus

Retrieves the opt-in status of a particular Region.

Request Syntax

```
POST /getRegionOptStatus HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "RegionName": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: ^\d{12}$

Required: No

RegionName

Specifies the Region-code for a given Region name (for example, `af-south-1`). This function will return the status of whatever Region you pass into this parameter.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "RegionName": "string",
   "RegionOptStatus": "string"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

RegionName

The Region code that was passed in.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

RegionOptStatus

One of the potential statuses a Region can undergo (Enabled, Enabling, Disabled, Disabling, Enabled_By_Default).

Type: String

Valid Values: `ENABLED | ENABLING | DISABLING | DISABLED | ENABLED_BY_DEFAULT`

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2

- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

ListRegions

Lists all the Regions for a given account and their respective opt-in statuses. Optionally, this list can be filtered by the `region-opt-status-contains` parameter.

Request Syntax

```
POST /listRegions HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "MaxResults": number,
   "NextToken": "string",
   "RegionOptStatusContains": [ "string" ]
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

MaxResults

The total number of items to return in the command's output. If the total number of items available is more than the value specified, a `NextToken` is provided in the command's output. To resume pagination, provide the `NextToken` value in the `starting-token` argument of a subsequent command. Do not use the `NextToken` response element directly outside of the AWS CLI. For usage examples, see Pagination in the *AWS Command Line Interface User Guide*.

Type: Integer

Valid Range: Minimum value of 1. Maximum value of 50.

Required: No

NextToken

A token used to specify where to start paginating. This is the `NextToken` from a previously truncated response. For usage examples, see Pagination in the *AWS Command Line Interface User Guide*.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 1000.

Required: No

RegionOptStatusContains

A list of Region statuses (Enabling, Enabled, Disabling, Disabled, Enabled_by_default) to use to filter the list of Regions for a given account. For example, passing in a value of ENABLING will only return a list of Regions with a Region status of ENABLING.

Type: Array of strings

Valid Values: `ENABLED | ENABLING | DISABLING | DISABLED | ENABLED_BY_DEFAULT`

Required: No

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "NextToken": "string",
   "Regions": [
      {
         "RegionName": "string",
         "RegionOptStatus": "string"
      }
   ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken

If there is more data to be returned, this will be populated. It should be passed into the `next-token` request parameter of `list-regions`.

Type: String

Regions

This is a list of Regions for a given account, or if the filtered parameter was used, a list of Regions that match the filter criteria set in the `filter` parameter.

Type: Array of Region objects

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

PutAlternateContact

Modifies the specified alternate contact attached to an AWS account.

For complete details about how to use the alternate contact operations, see Access or updating the alternate contacts.

> **Note**
>
> Before you can update the alternate contact information for an AWS account that is managed by AWS Organizations, you must first enable integration between AWS Account Management and Organizations. For more information, see Enabling trusted access for AWS Account Management.

Request Syntax

```
POST /putAlternateContact HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "AlternateContactType": "string",
   "EmailAddress": "string",
   "Name": "string",
   "PhoneNumber": "string",
   "Title": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12 digit account ID number of the AWS account that you want to access or modify with this operation.

If you do not specify this parameter, it defaults to the AWS account of the identity used to call the operation.

To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account, and the specified account ID must be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`; it must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, then don't specify this parameter, and call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

AlternateContactType

Specifies which alternate contact you want to create or update.

Type: String

Valid Values: `BILLING | OPERATIONS | SECURITY`

Required: Yes

EmailAddress

Specifies an email address for the alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `^[\s]*[\w+=.#!&-]+@[\w.-]+\.[\w]+[\s]*$`

Required: Yes

Name

Specifies a name for the alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Required: Yes

PhoneNumber

Specifies a phone number for the alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 25.

Pattern: `^[\s0-9()+-]+$`

Required: Yes

Title

Specifies a title for the alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

Examples

Example 1

The following example sets the billing alternate contact for the account whose credentials are used to call the operation.

Sample Request

```
POST / HTTP/1.1
X-Amz-Target: AWSAccountV20210201.PutAlternateContact

{
    "AlternateContactType": "Billing",
    "Name": "Carlos Salazar",
```

```
    "Title": "CFO",
    "EmailAddress": "carlos@example.com",
    "PhoneNumber": "206-555-0199"
}
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json
```

Example 2

The following example sets or overwrites the billing alternate contact for the specified member account in an organization. You must use credentials from either the organization's management account or from the Account Management service's delegated admin account.

Sample Request

```
POST / HTTP/1.1
X-Amz-Target: AWSAccountV20210201.PutAlternateContact

{
    "AccountId": "123456789012",
    "AlternateContactType": "Billing",
    "Name": "Carlos Salazar",
    "Title": "CFO",
    "EmailAddress": "carlos@example.com",
    "PhoneNumber": "206-555-0199"
}
```

Sample Response

```
HTTP/1.1 200 OK
Content-Type: application/json
```

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

PutContactInformation

Updates the primary contact information of an AWS account.

For complete details about how to use the primary contact operations, see Update the primary and alternate contact information.

Request Syntax

```
POST /putContactInformation HTTP/1.1
Content-type: application/json

{
   "AccountId": "string",
   "ContactInformation": {
      "AddressLine1": "string",
      "AddressLine2": "string",
      "AddressLine3": "string",
      "City": "string",
      "CompanyName": "string",
      "CountryCode": "string",
      "DistrictOrCounty": "string",
      "FullName": "string",
      "PhoneNumber": "string",
      "PostalCode": "string",
      "StateOrRegion": "string",
      "WebsiteUrl": "string"
   }
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

AccountId

Specifies the 12-digit account ID number of the AWS account that you want to access or modify with this operation. If you don't specify this parameter, it defaults to the AWS account of the

identity used to call the operation. To use this parameter, the caller must be an identity in the organization's management account or a delegated administrator account. The specified account ID must also be a member account in the same organization. The organization must have all features enabled, and the organization must have trusted access enabled for the Account Management service, and optionally a delegated admin account assigned.

> **Note**
>
> The management account can't specify its own `AccountId`. It must call the operation in standalone context by not including the `AccountId` parameter.

To call this operation on an account that is not a member of an organization, don't specify this parameter. Instead, call the operation using an identity belonging to the account whose contacts you wish to retrieve or modify.

Type: String

Pattern: `^\d{12}$`

Required: No

ContactInformation

Contains the details of the primary contact information associated with an AWS account.

Type: ContactInformation object

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

For information about the errors that are common to all actions, see Common Errors.

AccessDeniedException

The operation failed because the calling identity doesn't have the minimum required permissions.

HTTP Status Code: 403

InternalServerException

The operation failed because of an error internal to AWS. Try your operation again later.

HTTP Status Code: 500

TooManyRequestsException

The operation failed because it was called too frequently and exceeded a throttle limit.

HTTP Status Code: 429

ValidationException

The operation failed because one of the input parameters was invalid.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

Related actions in other AWS services

The following operations are related to AWS Account Management but are part of the AWS Organizations namespace:

- CreateAccount
- CreateGovCloudAccount
- DescribeAccount

CreateAccount

The `CreateAccount` API operation is available to use only in the context of an organization that is managed by the AWS Organizations service. The API operation is defined in that service's namespace.

For more information, see CreateAccount in the *AWS Organizations API Reference*.

CreateGovCloudAccount

The `CreateGovCloudAccount` API operation is available to use only in the context of an organization that is managed by the AWS Organizations service. The API operation is defined in that service's namespace.

For more information, see CreateGovCloudAccount in the *AWS Organizations API Reference*.

DescribeAccount

The `DescribeAccount` API operation is available to use only in the context of an organization that is managed by the AWS Organizations service. The API operation is defined in that service's namespace.

For more information, see DescribeAccount in the *AWS Organizations API Reference*.

Data Types

The following data types are supported:

- AlternateContact

- ContactInformation
- Region
- ValidationExceptionField

AlternateContact

A structure that contains the details of an alternate contact associated with an AWS account

Contents

AlternateContactType

The type of alternate contact.

Type: String

Valid Values: `BILLING | OPERATIONS | SECURITY`

Required: No

EmailAddress

The email address associated with this alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Pattern: `^[\s]*[\w+=.#!&-]+@[\w.-]+\.[\w]+[\s]*$`

Required: No

Name

The name associated with this alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 64.

Required: No

PhoneNumber

The phone number associated with this alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 25.

Pattern: `^[\s0-9()+-]+$`

Required: No

Title

The title associated with this alternate contact.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ContactInformation

Contains the details of the primary contact information associated with an AWS account.

Contents

AddressLine1

The first line of the primary contact address.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 60.

Required: Yes

City

The city of the primary contact address.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

CountryCode

The ISO-3166 two-letter country code for the primary contact address.

Type: String

Length Constraints: Fixed length of 2.

Required: Yes

FullName

The full name of the primary contact address.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: Yes

PhoneNumber

The phone number of the primary contact information. The number will be validated and, in some countries, checked for activation.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 20.

Pattern: `^[+][\s0-9()-]+$`

Required: Yes

PostalCode

The postal code of the primary contact address.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 20.

Required: Yes

AddressLine2

The second line of the primary contact address, if any.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 60.

Required: No

AddressLine3

The third line of the primary contact address, if any.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 60.

Required: No

CompanyName

The name of the company associated with the primary contact information, if any.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: No

DistrictOrCounty

The district or county of the primary contact address, if any.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: No

StateOrRegion

The state or region of the primary contact address. If the mailing address is within the United States (US), the value in this field can be either a two character state code (for example, `NJ`) or the full state name (for example, `New Jersey`). This field is required in the following countries: `US`, `CA`, `GB`, `DE`, `JP`, `IN`, and `BR`.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: No

WebsiteUrl

The URL of the website associated with the primary contact information, if any.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 256.

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Region

This is a structure that expresses the Region for a given account, consisting of a name and opt-in status.

Contents

RegionName

The Region code of a given Region (for example, `us-east-1`).

Type: String

Length Constraints: Minimum length of 1. Maximum length of 50.

Required: No

RegionOptStatus

One of potential statuses a Region can undergo (Enabled, Enabling, Disabled, Disabling, Enabled_By_Default).

Type: String

Valid Values: `ENABLED | ENABLING | DISABLING | DISABLED | ENABLED_BY_DEFAULT`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ValidationExceptionField

The input failed to meet the constraints specified by the AWS service in a specified field.

Contents

message

A message about the validation exception.

Type: String

Required: Yes

name

The field name where the invalid entry was detected.

Type: String

Required: Yes

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Common Parameters

The following list contains the parameters that all actions use for signing Signature Version 4 requests with a query string. Any action-specific parameters are listed in the topic for that action. For more information about Signature Version 4, see Signing AWS API requests in the *IAM User Guide*.

Action

The action to be performed.

Type: string

Required: Yes

Version

The API version that the request is written for, expressed in the format YYYY-MM-DD.

Type: string

Required: Yes

X-Amz-Algorithm

The hash algorithm that you used to create the request signature.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Valid Values: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

The credential scope value, which is a string that includes your access key, the date, the region you are targeting, the service you are requesting, and a termination string ("aws4_request"). The value is expressed in the following format: *access_key*/*YYYYMMDD*/*region*/*service*/aws4_request.

For more information, see Create a signed AWS API request in the *IAM User Guide*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-Date

The date that is used to create the signature. The format must be ISO 8601 basic format (YYYYMMDD'T'HHMMSS'Z'). For example, the following date time is a valid X-Amz-Date value: `20120325T120000Z`.

Condition: X-Amz-Date is optional for all requests; it can be used to override the date used for signing requests. If the Date header is specified in the ISO 8601 basic format, X-Amz-Date is not required. When X-Amz-Date is used, it always overrides the value of the Date header. For more information, see Elements of an AWS API request signature in the *IAM User Guide*.

Type: string

Required: Conditional

X-Amz-Security-Token

The temporary security token that was obtained through a call to AWS Security Token Service (AWS STS). For a list of services that support temporary security credentials from AWS STS, see AWS services that work with IAM in the *IAM User Guide*.

Condition: If you're using temporary security credentials from AWS STS, you must include the security token.

Type: string

Required: Conditional

X-Amz-Signature

Specifies the hex-encoded signature that was calculated from the string to sign and the derived signing key.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

X-Amz-SignedHeaders

Specifies all the HTTP headers that were included as part of the canonical request. For more information about specifying signed headers, see Create a signed AWS API request in the *IAM User Guide*.

Condition: Specify this parameter when you include authentication information in a query string instead of in the HTTP authorization header.

Type: string

Required: Conditional

Common Errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

NotAuthorized

You do not have permission to perform this action.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Calling the API by making HTTP Query requests

This section contains general information about using the Query API for AWS Account Management. For details about the API operations and errors, see the API Reference.

> **Note**
>
> Instead of making direct calls to the AWS Account Management Query API, you can use one of the AWS SDKs. The AWS SDKs consist of libraries and sample code for various programming languages and platforms (Java, Ruby, .NET, iOS, Android, and more). The SDKs provide a convenient way to create programmatic access to AWS Account Management and AWS. For example, the SDKs take care of tasks such as cryptographically

signing requests, managing errors, and retrying requests automatically. For information about the AWS SDKs, including how to download and install them, see Tools for Amazon Web Services.

With the Query API for AWS Account Management, you can call service actions. Query API requests are HTTPS requests that must contain an `Action` parameter to indicate the operation to be performed. AWS Account Management supports `GET` and `POST` requests for all operations. That is, the API doesn't require you to use `GET` for some actions and `POST` for others. However, `GET` requests are subject to the limitation size of a URL. Although this limit is browser dependent, a typical limit is 2,048 bytes. Therefore, for Query API requests that require larger sizes, you must use a `POST` request.

The response is an XML document. For details about the response, see the individual action pages in the API Reference.

Topics

Endpoints

AWS Account Management has a single global API endpoint that is hosted in the US East (N. Virginia) AWS Region.

For more information about AWS endpoints and Regions for all services, see Regions and Endpoints in the *AWS General Reference*.

HTTPS required

Because the Query API can return sensitive information such as security credentials, you must use HTTPS to encrypt all API requests.

Signing AWS Account Management API requests

Requests must be signed using an access key ID and a secret access key. We strongly recommend that you don't use your AWS root account credentials for everyday work with AWS Account

Management. You can use the credentials for an AWS Identity and Access Management (IAM) user or temporary credentials such as you use with an IAM role.

To sign your API requests, you must use AWS Signature Version 4. For information about using Signature Version 4, see Signing AWS API requests in the *IAM User Guide*.

For more information, see the following:

- AWS Security Credentials – Provides general information about the types of credentials that you can use to access AWS.
- Security best practices in IAM – Offers suggestions for using the IAM service to help secure your AWS resources, including those in AWS Account Management.
- Temporary security credentials in IAM – Describes how to create and use temporary security credentials.

Quotas for AWS Account Management

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is AWS Region-specific.

Each AWS account has the following quotas related to Account Management.

Resource	Quota
Number of alternate contacts in an AWS account	3 - one each for `BILLING`, `SECURITY`, and `OPERATIONS`
Number of concurrent region-opt requests per account	6
Number of concurrent region-opt requests per organization	20
Rate of `DeleteAlternateContact` requests per account	1 per second, burst to 6 per second
Rate of `DisableRegion` requests per account	1 per second, burst to 1 per second
Rate of `EnableRegion` requests per account	1 per second, burst to 1 per second
Rate of `GetAlternateContact` requests per account	10 per second, burst to 15 per second
Rate of `GetContactInformation` requests per account	10 per second, burst to 15 per second
Rate of `GetRegionOptStatus` requests per account	5 per second, burst to 5 per second
Rate of `ListRegions` requests per account	5 per second, burst to 5 per second
Rate of `PutAlternateContact` requests per account	5 per second, burst to 8 per second

Resource	Quota
Rate of `PutContactInformation` requests per account	5 per second, burst to 8 per second

Troubleshooting your AWS account

Use the information in the following topics to help you diagnose and fix issues with your AWS account. For help with the root user, see Troubleshooting issues with the root user in the *IAM User Guide*. For help with the sign-in process, see Troubleshooting AWS account sign-in issues in the *AWS Sign-In User Guide*.

Troubleshooting topics

- Troubleshooting issues with AWS account creation
- Troubleshooting issues with AWS account closure
- Troubleshooting other issues with AWS accounts

Troubleshooting issues with AWS account creation

Use the reference links in the following table to help you diagnose and fix issues with creating a new AWS account.

Issue	Reference link	Source
I don't know how to sign-up or create an account	Create a standalone AWS account	This guide
What should I do if I didn't receive a call from AWS to verify my new account or the PIN I entered doesn't work?	https://repost.aws/knowledge-center/phone-verify-no-call	AWS re:Post
How do I resolve the "maximum number of failed attempts" error when I try to verify my AWS account by phone?	https://repost.aws/knowledge-center/maximum-failed-attempts	AWS re:Post
It's been more than 24 hours and my account isn't activated	https://repost.aws/knowledge-center/create-and-activate-aws-account	AWS re:Post

Issue	Reference link	Source
I can't sign into my new account after it has been created	https://docs.aws.amazon.com/signin/latest/userguide/troubleshooting-sign-in-issues.html	AWS Sign-In User Guide

For additional help, we recommend that you search AWS re:Post for content related to your specific issue. If you still need assistance, contact AWS Support.

Troubleshooting issues with AWS account closure

Use the information below to help you diagnose and fix common issues found during the account closure process. For general information about the account closure process, see Close an AWS account.

Topics

- I don’t know how to delete or cancel my account
- I don’t see the Close account button on the Accounts page
- I closed my account but still haven’t received an email confirmation
- I receive a "ConstraintViolationException" error when trying to close my account
- I receive a "CLOSE_ACCOUNT_QUOTA_EXCEEDED" error when trying to close a member account
- Do I need to delete my AWS organization before closing the management account?

I don’t know how to delete or cancel my account

To close your account, follow the instructions in Close an AWS account.

I don’t see the Close account button on the Accounts page

If you are not signed in as the root user, you will not see the **Close account** button displayed on the **Accounts** page. You must Sign in to the AWS Management Console as the root user to close your account. If you can’t sign in, see Troubleshooting issues with the root user.

I closed my account but still haven't received an email confirmation

This confirmation email is only sent to the root user email address for the AWS account. If you don't receive this email within a few hours, you can Sign in to the AWS Management Console as the root user to check that your account is closed. If your account was closed successfully, you will see a message displayed that indicates your account is closed. If the account you closed is a member account, you can verify successful closure by checking if the closed account is labeled as SUSPENDED in the AWS Organizations console. For more information, see Closing a member account in your organization in the *AWS Organizations User Guide*.

If you are trying to close a **management account** and do not receive an email confirmation about the account closure, your organization most likely has active member accounts. You can only close the management account if your organization doesn't have any active member accounts. To verify that there are no active member accounts remaining in your organization, go to the AWS Organizations console, and make sure that all member accounts are showing Suspended next to their account names. After that, you can close the management account.

I receive a "ConstraintViolationException" error when trying to close my account

You are trying to close a management account using the AWS Organizations console, which is not possible. To close a management account, you need to Sign in to the AWS Management Console as the root user for the management account and close it from the **Accounts** page. For more information, see Closing a management account in your organization in the *AWS Organizations User Guide*.

I receive a "CLOSE_ACCOUNT_QUOTA_EXCEEDED" error when trying to close a member account

You can only close 10% of member accounts within a rolling 30 day period. This quota is not bound by a calendar month, but starts when you close an account. Within 30 days of that initial account closure, you can't exceed the 10% account closure limit. The minimum account closure is 10 and the maximum account closure is 1000, even if 10% of accounts exceeds 1000. For more information about Organizations quotas, see Quotas for AWS Organizations in the *AWS Organizations User Guide*.

Do I need to delete my AWS organization before closing the management account?

No, you don't need to delete your AWS organization before closing the management account. However, you can only close the management account if your organization doesn't have any active member accounts. To verify that there are no active member accounts remaining in your organization, go to the AWS Organizations console, and make sure that all member accounts are showing Suspended next to their account names. After that, you can close the management account.

Troubleshooting other issues with AWS accounts

Use the information here to help you troubleshoot issues related to your AWS account.

Issues

- I need to change the credit card for my AWS account
- I need to report fraudulent AWS account activity
- I need to close my AWS account

I need to change the credit card for my AWS account

To change the credit card for your AWS account, you must be able to sign in. AWS has protections in place that require you to prove that you're the account owner. For instructions, see Managing your credit card payment methods in the *AWS Billing User Guide*.

I need to report fraudulent AWS account activity

If you suspect fraudulent activity using your AWS account and would like to make a report, see How do I report abuse of AWS resources.

If you're having trouble with a purchase made on Amazon.com, see Amazon Customer Service.

I need to close my AWS account

For help troubleshooting issues with closing your AWS account, see Close an AWS account.

Document history for the Account Management User Guide

The following table describes the documentation releases for AWS Account Management.

Change	Description	Date
Rewrite of the close account topic	Completely overhauled the entire close account topic including adding steps for how to close member and management accounts.	February 1, 2024
End of support for adding new security challenge questions	Added new content noting that the option to add new challenge questions was removed from the **Accounts** page.	January 5, 2024
End of support for the `aws-portal` namespace	AWS Identity and Access Management (IAM) actions that were previously used to manage your account (for example, `aws-portal:ModifyAccount` and `aws-portal:ViewAccount`) have reached the end of standard support.	January 1, 2024
Rewrite of the Regions topic	Completely overhauled the entire Regions topic including adding expand and collapse controls.	October 8, 2023
Relocated root user topics to the IAM User Guide	Consolidated discussion about root users into one topic, added cross reference links	September 18, 2023

	to root user topics that were moved to the IAM User Guide.	
New section added to the primary account contact topic	Added new *Phone number and email address requirements* section.	September 12, 2023
New contact information APIs	Support for new `GetContactInformation` and `PutContactInformation` APIs.	July 22, 2022
AWS Account Management now supports updating alternate contacts via the AWS Organizations console.	You can now update your organization's alternate contacts via AWS Organizations console using Account API permissions provided by updated AWS Organizations managed policies.	February 8, 2022
Initial release	Initial release of the AWS Account Management Reference Guide	September 30, 2021

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the *AWS Glossary Reference*.