API Reference

AWS Security Token Service

API Version 2011-06-15

AWS Security Token Service: API Reference

Table of Contents

Welcome to the AWS Security Token Service API Reference

AWS provides AWS Security Token Service (AWS STS) as a web service that enables you to request temporary, limited-privilege credentials for users. This guide describes the AWS STS API. For more information, see Temporary Security Credentials in the *IAM User Guide*.

> **Note**
>
> As an alternative to using the API, you can use one of the AWS SDKs, which consist of libraries and sample code for various programming languages and platforms such as Java, Ruby, .NET, iOS, Android, and others. The SDKs provide a convenient way to create programmatic access to AWS STS. For example, the SDKs can cryptographically sign requests, manage errors, and retry requests automatically. For information about the AWS SDKs, see Tools to Build on AWS.

For information about setting up signatures and authorization through the API, see Signing AWS API Requests in the *Amazon Web Services General Reference*. For general information about the Query API, see Making Query Requests in the *IAM User Guide*. For information about using security tokens with other AWS products, see AWS Services That Work with IAM in the *IAM User Guide*.

Endpoints

By default, AWS Security Token Service (AWS STS) is available as a global service, and all AWS STS requests go to a single endpoint at `https://sts.amazonaws.com`. Global requests map to the US East (N. Virginia) Region. AWS recommends using Regional AWS STS endpoints instead of the global endpoint to reduce latency, build in redundancy, and increase session token validity. For more information, see Managing AWS STS in an AWS Region in the *IAM User Guide*.

Most AWS Regions enable operations in all AWS services by default. These Regions automatically activate for use with AWS STS. Some Regions, such as Asia Pacific (Hong Kong), must be manually enabled. To learn more about enabling and disabling AWS Regions, see Managing AWS Regions in the *Amazon Web Services General Reference*. When you enable these AWS Regions, they are automatically activated for use with AWS STS. You cannot activate the AWS STS endpoint for a disabled Region. Tokens that are valid in all AWS Regions include more characters than tokens that

are valid in Regions enabled by default. Changing this setting might affect existing systems where you temporarily store tokens. For more information, see Managing Global Endpoint Session Tokens in the *IAM User Guide*.

After you activate a Region for use with AWS STS, you can direct AWS STS API calls to that Region. AWS STS recommends you provide both the Region and endpoint when you send calls to a Regional endpoint. You can provide the Region alone for manually enabled Regions, such as Asia Pacific (Hong Kong). In this case, you direct the calls to the AWS STS Regional endpoint. However, if you provide the Region alone for Regions enabled by default, AWS STS directs the calls to the global endpoint of `https://sts.amazonaws.com`.

To view the list of AWS STS endpoints and if they are active by default, see Writing Code to Use AWS STS Regions in the *IAM User Guide*.

Recording API requests

AWS STS supports AWS CloudTrail, a service that records AWS calls for your AWS account and delivers log files to an Amazon S3 bucket. By using information collected by CloudTrail, you can determine the requests successfully sent to AWS STS, as well as who sent the request, and when it was sent. For more information about CloudTrail, including how to enable it and find your log files, see Logging IAM and AWS STS API calls with AWS CloudTrail in the *IAM User Guide* and the AWS CloudTrail User Guide.

Actions

The following actions are supported:

- AssumeRole
- AssumeRoleWithSAML
- AssumeRoleWithWebIdentity
- DecodeAuthorizationMessage
- GetAccessKeyInfo
- GetCallerIdentity
- GetFederationToken
- GetSessionToken

AssumeRole

Returns a set of temporary security credentials that you can use to access AWS resources. These temporary credentials consist of an access key ID, a secret access key, and a security token. Typically, you use `AssumeRole` within your account or for cross-account access. For a comparison of `AssumeRole` with other API operations that produce temporary credentials, see Requesting Temporary Security Credentials and Comparing the AWS STS API operations in the *IAM User Guide*.

Permissions

The temporary security credentials created by `AssumeRole` can be used to make API calls to any AWS service with the following exception: You cannot call the AWS STS `GetFederationToken` or `GetSessionToken` API operations.

(Optional) You can pass inline or managed session policies to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies. The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

When you create a role, you create two policies: a role trust policy that specifies *who* can assume the role, and a permissions policy that specifies *what* can be done with the role. You specify the trusted principal that is allowed to assume the role in the role trust policy.

To assume a role from a different account, your AWS account must be trusted by the role. The trust relationship is defined in the role's trust policy when the role is created. That trust policy states which accounts are allowed to delegate that access to users in the account.

A user who wants to access a role in a different account must also have permissions that are delegated from the account administrator. The administrator must attach a policy that allows the user to call `AssumeRole` for the ARN of the role in the other account.

To allow a user to assume a role in the same account, you can do either of the following:

- Attach a policy to the user that allows the user to call `AssumeRole` (as long as the role's trust policy trusts the account).
- Add the user as a principal directly in the role's trust policy.

You can do either because the role's trust policy acts as an IAM resource-based policy. When a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information about trust policies and resource-based policies, see IAM Policies in the *IAM User Guide*.

Tags

(Optional) You can pass tag key-value pairs to your session. These tags are called session tags. For more information about session tags, see Passing Session Tags in AWS STS in the *IAM User Guide*.

An administrator must grant you the permissions necessary to pass session tags. The administrator can also create granular permissions to allow you to pass only specific session tags. For more information, see Tutorial: Using Tags for Attribute-Based Access Control in the *IAM User Guide*.

You can set the session tags as transitive. Transitive tags persist during role chaining. For more information, see Chaining Roles with Session Tags in the *IAM User Guide*.

Using MFA with AssumeRole

(Optional) You can include multi-factor authentication (MFA) information when you call `AssumeRole`. This is useful for cross-account scenarios to ensure that the user that assumes the role has been authenticated with an AWS MFA device. In that scenario, the trust policy of the role being assumed includes a condition that tests for MFA authentication. If the caller does not include valid MFA information, the request to assume the role is denied. The condition in a trust policy that tests for MFA authentication might look like the following example.

```
"Condition": {"Bool": {"aws:MultiFactorAuthPresent": true}}
```

For more information, see Configuring MFA-Protected API Access in the *IAM User Guide* guide.

To use MFA with `AssumeRole`, you pass values for the `SerialNumber` and `TokenCode` parameters. The `SerialNumber` value identifies the user's hardware or virtual MFA device. The `TokenCode` is the time-based one-time password (TOTP) that the MFA device produces.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

DurationSeconds

The duration, in seconds, of the role session. The value specified can range from 900 seconds (15 minutes) up to the maximum session duration set for the role. The maximum session duration setting can have a value from 1 hour to 12 hours. If you specify a value higher than this setting or the administrator setting (whichever is lower), the operation fails. For example, if you specify a session duration of 12 hours, but your administrator set the maximum session duration to 6 hours, your operation fails.

Role chaining limits your AWS CLI or AWS API role session to a maximum of one hour. When you use the `AssumeRole` API operation to assume a role, you can specify the duration of your role session with the `DurationSeconds` parameter. You can specify a parameter value of up to 43200 seconds (12 hours), depending on the maximum session duration setting for your role. However, if you assume a role using role chaining and provide a `DurationSeconds` parameter value greater than one hour, the operation fails. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*.

By default, the value is set to `3600` seconds.

> **Note**
>
> The `DurationSeconds` parameter is separate from the duration of a console session that you might request using the returned credentials. The request to the federation endpoint for a console sign-in token takes a `SessionDuration` parameter that specifies the maximum length of the console session. For more information, see Creating a URL that Enables Federated Users to Access the AWS Management Console in the *IAM User Guide*.

Type: Integer

Valid Range: Minimum value of 900. Maximum value of 43200.

Required: No

ExternalId

A unique identifier that might be required when you assume a role in another account. If the administrator of the account to which the role belongs provided you with an external ID, then provide that value in the `ExternalId` parameter. This value can be any string, such as a passphrase or account number. A cross-account role is usually set up to trust everyone in an

account. Therefore, the administrator of the trusting account might send an external ID to the administrator of the trusted account. That way, only someone with the ID can assume the role, rather than everyone in the account. For more information about the external ID, see How to Use an External ID When Granting Access to Your AWS Resources to a Third Party in the *IAM User Guide*.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@:/-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 1224.

Pattern: `[\w+=,.@:\/-]*`

Required: No

Policy

An IAM policy in JSON format that you want to use as an inline session policy.

This parameter is optional. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. The JSON policy characters can be any ASCII character from the space character to the end of the valid character list (\u0020 through \u00FF). It can also include the tab (\u0009), linefeed (\u000A), and carriage return (\u000D) characters.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Type: String

Length Constraints: Minimum length of 1.

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]+`

Required: No

PolicyArns.member.N

The Amazon Resource Names (ARNs) of the IAM managed policies that you want to use as managed session policies. The policies must exist in the same account as the role.

This parameter is optional. You can provide up to 10 managed policy ARNs. However, the plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. For more information about ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the AWS General Reference.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

Type: Array of PolicyDescriptorType objects

Required: No

ProvidedContexts.member.N

A list of previously acquired trusted context assertions in the format of a JSON array. The trusted context assertion is signed and encrypted by AWS STS.

The following is an example of a `ProvidedContext` value that includes a single trusted context assertion and the ARN of the context provider from which the trusted context assertion was generated.

```
[{"ProviderArn":"arn:aws:iam::aws:contextProvider/
IdentityCenter","ContextAssertion":"trusted-context-assertion"}]
```

Type: Array of ProvidedContext objects

Array Members: Maximum number of 5 items.

Required: No

RoleArn

The Amazon Resource Name (ARN) of the role to assume.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

RoleSessionName

An identifier for the assumed role session.

Use the role session name to uniquely identify a session when the same role is assumed by different principals or for different reasons. In cross-account scenarios, the role session name is visible to, and can be logged by the account that owns the role. The role session name is also used in the ARN of the assumed role principal. This means that subsequent cross-account API requests that use the temporary security credentials will expose the role session name to the external account in their AWS CloudTrail logs.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

Required: Yes

SerialNumber

The identification number of the MFA device that is associated with the user who is making the `AssumeRole` call. Specify this value if the trust policy of the role being assumed includes a condition that requires MFA authentication. The value is either the serial number for a hardware device (such as `GAHT12345678`) or an Amazon Resource Name (ARN) for a virtual device (such as `arn:aws:iam::123456789012:mfa/user`).

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 9. Maximum length of 256.

Pattern: `[\w+=/:,.@-]*`

Required: No

SourceIdentity

The source identity specified by the principal that is calling the `AssumeRole` operation.

You can require users to specify a source identity when they assume a role. You do this by using the `sts:SourceIdentity` condition key in a role trust policy. You can use source identity information in AWS CloudTrail logs to determine who took actions with a role. You can use the `aws:SourceIdentity` condition key to further control access to AWS resources based on the value of source identity. For more information about using source identity, see Monitor and control actions taken with assumed roles in the *IAM User Guide*.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-. You cannot use a value that begins with the text `aws:`. This prefix is reserved for AWS internal use.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

Required: No

Tags.member.N

A list of session tags that you want to pass. Each session tag consists of a key name and an associated value. For more information about session tags, see Tagging AWS STS Sessions in the *IAM User Guide*.

This parameter is optional. You can pass up to 50 session tags. The plaintext session tag keys can't exceed 128 characters, and the values can't exceed 256 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

You can pass a session tag with the same key as a tag that is already attached to the role. When you do, session tags override a role tag with the same key.

Tag key–value pairs are not case sensitive, but case is preserved. This means that you cannot have separate `Department` and `department` tag keys. Assume that the role has the `Department=Marketing` tag and you pass the `department=engineering` session tag. `Department` and `department` are not saved as separate tags, and the session tag passed in the request takes precedence over the role tag.

Additionally, if you used temporary credentials to perform this operation, the new session inherits any transitive session tags from the calling session. If you pass a session tag with the same key as an inherited tag, the operation fails. To view the inherited tags for a session, see the AWS CloudTrail logs. For more information, see Viewing Session Tags in CloudTrail in the *IAM User Guide*.

Type: Array of Tag objects

Array Members: Maximum number of 50 items.

Required: No

TokenCode

The value provided by the MFA device, if the trust policy of the role being assumed requires MFA. (In other words, if the policy includes a condition that tests for MFA). If the role being assumed requires MFA and if the `TokenCode` value is missing or expired, the `AssumeRole` call returns an "access denied" error.

The format for this parameter, as described by its regex pattern, is a sequence of six numeric digits.

Type: String

Length Constraints: Fixed length of 6.

Pattern: `[\d]*`

Required: No

TransitiveTagKeys.member.N

A list of keys for session tags that you want to set as transitive. If you set a tag key as transitive, the corresponding key and value passes to subsequent sessions in a role chain. For more information, see Chaining Roles with Session Tags in the *IAM User Guide*.

This parameter is optional. When you set session tags as transitive, the session policy and session tags packed binary limit is not affected.

If you choose not to specify a transitive tag key, then no tags are passed from this session to any subsequent sessions.

Type: Array of strings

Array Members: Maximum number of 50 items.

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[\p{L}\p{Z}\p{N}_.:/=+\-@]+`

Required: No

Response Elements

The following elements are returned by the service.

AssumedRoleUser

The Amazon Resource Name (ARN) and the assumed role ID, which are identifiers that you can use to refer to the resulting temporary security credentials. For example, you can reference these credentials as a principal in a resource-based policy by using the ARN or assumed role ID. The ARN and ID include the `RoleSessionName` that you specified when you called `AssumeRole`.

Type: AssumedRoleUser object

Credentials

The temporary security credentials, which include an access key ID, a secret access key, and a security (or session) token.

> **Note**
>
> The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size.

Type: Credentials object

PackedPolicySize

A percentage value that indicates the packed size of the session policies and session tags combined passed in the request. The request fails if the packed size is greater than 100 percent, which means the policies and tags exceeded the allowed space.

Type: Integer

Valid Range: Minimum value of 0.

SourceIdentity

The source identity specified by the principal that is calling the `AssumeRole` operation.

You can require users to specify a source identity when they assume a role. You do this by using the `sts:SourceIdentity` condition key in a role trust policy. You can use source identity information in AWS CloudTrail logs to determine who took actions with a role. You can use the `aws:SourceIdentity` condition key to further control access to AWS resources based on the value of source identity. For more information about using source identity, see Monitor and control actions taken with assumed roles in the *IAM User Guide*.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

Errors

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

ExpiredToken

The web identity token that was passed is expired or is not valid. Get a new identity token from the identity provider and then retry the request.

HTTP Status Code: 400

MalformedPolicyDocument

The request was rejected because the policy document was malformed. The error message describes the specific error.

HTTP Status Code: 400

PackedPolicyTooLarge

The request was rejected because the total packed size of the session policies and session tags combined was too large. An AWS conversion compresses the session policy document, session policy ARNs, and session tags into a packed binary format that has a separate limit. The error message indicates by percentage how close the policies and tags are to the upper size limit. For more information, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You could receive this error even though you meet other defined session policy and session tag limits. For more information, see IAM and AWS STS Entity Character Limits in the *IAM User Guide*.

HTTP Status Code: 400

RegionDisabled

AWS STS is not activated in the requested region for the account that is being asked to generate credentials. The account administrator must use the IAM console to activate AWS STS in that region. For more information, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

HTTP Status Code: 403

Examples

Example

This example illustrates one usage of AssumeRole.

Sample Request

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRole
&RoleSessionName=testAR
&RoleArn=arn:aws:iam::123456789012:role/demo
&PolicyArns.member.1.arn=arn:aws:iam::123456789012:policy/demopolicy1
&PolicyArns.member.2.arn=arn:aws:iam::123456789012:policy/demopolicy2
&Policy={"Version":"2012-10-17","Statement":[{"Sid":"Stmt1",
"Effect":"Allow","Action":"s3:*","Resource":"*"}]}
&DurationSeconds=3600
&Tags.member.1.Key=Project
&Tags.member.1.Value=Pegasus
&Tags.member.2.Key=Team
&Tags.member.2.Value=Engineering
&Tags.member.3.Key=Cost-Center
&Tags.member.3.Value=12345
&TransitiveTagKeys.member.1=Project
&TransitiveTagKeys.member.2=Cost-Center
&ExternalId=123ABC
```

```
&SourceIdentity=Alice
&AUTHPARAMS
```

Sample Response

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <AssumeRoleResult>
  <SourceIdentity>Alice</SourceIdentity>
    <AssumedRoleUser>
      <Arn>arn:aws:sts::123456789012:assumed-role/demo/TestAR</Arn>
      <AssumedRoleId>ARO123EXAMPLE123:TestAR</AssumedRoleId>
    </AssumedRoleUser>
    <Credentials>
      <AccessKeyId>ASIAIOSFODNN7EXAMPLE</AccessKeyId>
      <SecretAccessKey>wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY</SecretAccessKey>
      <SessionToken>
       AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
       LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
       QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
       9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
       +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
      </SessionToken>
      <Expiration>2019-11-09T13:34:41Z</Expiration>
    </Credentials>
    <PackedPolicySize>6</PackedPolicySize>
  </AssumeRoleResult>
  <ResponseMetadata>
    <RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
  </ResponseMetadata>
</AssumeRoleResponse>
```

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

AssumeRoleWithSAML

Returns a set of temporary security credentials for users who have been authenticated via a SAML authentication response. This operation provides a mechanism for tying an enterprise identity store or directory to role-based AWS access without user-specific credentials or configuration. For a comparison of `AssumeRoleWithSAML` with the other API operations that produce temporary credentials, see Requesting Temporary Security Credentials and Comparing the AWS STS API operations in the *IAM User Guide*.

The temporary security credentials returned by this operation consist of an access key ID, a secret access key, and a security token. Applications can use these temporary security credentials to sign calls to AWS services.

Session Duration

By default, the temporary security credentials created by `AssumeRoleWithSAML` last for one hour. However, you can use the optional `DurationSeconds` parameter to specify the duration of your session. Your role session lasts for the duration that you specify, or until the time specified in the SAML authentication response's `SessionNotOnOrAfter` value, whichever is shorter. You can provide a `DurationSeconds` value from 900 seconds (15 minutes) up to the maximum session duration setting for the role. This setting can have a value from 1 hour to 12 hours. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*. The maximum session duration limit applies when you use the `AssumeRole*` API operations or the `assume-role*` CLI commands. However the limit does not apply when you use those operations to create a console URL. For more information, see Using IAM Roles in the *IAM User Guide*.

> **Note**
>
> Role chaining limits your AWS CLI or AWS API role session to a maximum of one hour. When you use the `AssumeRole` API operation to assume a role, you can specify the duration of your role session with the `DurationSeconds` parameter. You can specify a parameter value of up to 43200 seconds (12 hours), depending on the maximum session duration setting for your role. However, if you assume a role using role chaining and provide a `DurationSeconds` parameter value greater than one hour, the operation fails.

Permissions

The temporary security credentials created by `AssumeRoleWithSAML` can be used to make API calls to any AWS service with the following exception: you cannot call the AWS STS `GetFederationToken` or `GetSessionToken` API operations.

(Optional) You can pass inline or managed session policies to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies. The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

Calling `AssumeRoleWithSAML` does not require the use of AWS security credentials. The identity of the caller is validated by using keys in the metadata document that is uploaded for the SAML provider entity for your identity provider.

Important

Calling `AssumeRoleWithSAML` can result in an entry in your AWS CloudTrail logs. The entry includes the value in the `NameID` element of the SAML assertion. We recommend that you use a `NameIDType` that is not associated with any personally identifiable information (PII). For example, you could instead use the persistent identifier (`urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`).

Tags

(Optional) You can configure your IdP to pass attributes into your SAML assertion as session tags. Each session tag consists of a key name and an associated value. For more information about session tags, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You can pass up to 50 session tags. The plaintext session tag keys can’t exceed 128 characters and the values can’t exceed 256 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

You can pass a session tag with the same key as a tag that is attached to the role. When you do, session tags override the role's tags with the same key.

An administrator must grant you the permissions necessary to pass session tags. The administrator can also create granular permissions to allow you to pass only specific session tags. For more information, see Tutorial: Using Tags for Attribute-Based Access Control in the *IAM User Guide*.

You can set the session tags as transitive. Transitive tags persist during role chaining. For more information, see Chaining Roles with Session Tags in the *IAM User Guide*.

SAML Configuration

Before your application can call `AssumeRoleWithSAML`, you must configure your SAML identity provider (IdP) to issue the claims required by AWS. Additionally, you must use AWS Identity and Access Management (IAM) to create a SAML provider entity in your AWS account that represents your identity provider. You must also create an IAM role that specifies this SAML provider in its trust policy.

For more information, see the following resources:

- About SAML 2.0-based Federation in the *IAM User Guide*.
- Creating SAML Identity Providers in the *IAM User Guide*.
- Configuring a Relying Party and Claims in the *IAM User Guide*.
- Creating a Role for SAML 2.0 Federation in the *IAM User Guide*.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

DurationSeconds

The duration, in seconds, of the role session. Your role session lasts for the duration that you specify for the `DurationSeconds` parameter, or until the time specified in the SAML authentication response's `SessionNotOnOrAfter` value, whichever is shorter. You can provide a `DurationSeconds` value from 900 seconds (15 minutes) up to the maximum session duration setting for the role. This setting can have a value from 1 hour to 12 hours. If you specify a value higher than this setting, the operation fails. For example, if you specify a session duration of 12 hours, but your administrator set the maximum session duration to 6 hours, your operation fails. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*.

By default, the value is set to `3600` seconds.

> **Note**
>
> The `DurationSeconds` parameter is separate from the duration of a console session that you might request using the returned credentials. The request to the federation endpoint for a console sign-in token takes a `SessionDuration` parameter that specifies the maximum length of the console session. For more information, see Creating a URL that Enables Federated Users to Access the AWS Management Console in the *IAM User Guide*.

Type: Integer

Valid Range: Minimum value of 900. Maximum value of 43200.

Required: No

Policy

An IAM policy in JSON format that you want to use as an inline session policy.

This parameter is optional. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. The JSON policy characters can be any ASCII character from the space character to the end of the valid character list (\u0020 through \u00FF). It can also include the tab (\u0009), linefeed (\u000A), and carriage return (\u000D) characters.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]+`

Required: No

PolicyArns.member.N

The Amazon Resource Names (ARNs) of the IAM managed policies that you want to use as managed session policies. The policies must exist in the same account as the role.

This parameter is optional. You can provide up to 10 managed policy ARNs. However, the plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. For more information about ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the AWS General Reference.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

Type: Array of PolicyDescriptorType objects

Required: No

PrincipalArn

The Amazon Resource Name (ARN) of the SAML provider in IAM that describes the IdP.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

RoleArn

The Amazon Resource Name (ARN) of the role that the caller is assuming.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

SAMLAssertion

The base64 encoded SAML authentication response provided by the IdP.

For more information, see Configuring a Relying Party and Adding Claims in the *IAM User Guide*.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 100000.

Required: Yes

Response Elements

The following elements are returned by the service.

AssumedRoleUser

The identifiers for the temporary security credentials that the operation returns.

Type: AssumedRoleUser object

Audience

The value of the `Recipient` attribute of the `SubjectConfirmationData` element of the SAML assertion.

Type: String

Credentials

The temporary security credentials, which include an access key ID, a secret access key, and a security (or session) token.

> **Note**
>
> The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size.

Type: Credentials object

Issuer

The value of the `Issuer` element of the SAML assertion.

Type: String

NameQualifier

A hash value based on the concatenation of the following:

- The `Issuer` response value.
- The AWS account ID.
- The friendly name (the last part of the ARN) of the SAML provider in IAM.

The combination of `NameQualifier` and `Subject` can be used to uniquely identify a user.

The following pseudocode shows how the hash value is calculated:

```
BASE64 ( SHA1 ( "https://example.com/saml" + "123456789012" + "/
MySAMLIdP" ) )
```

Type: String

PackedPolicySize

A percentage value that indicates the packed size of the session policies and session tags combined passed in the request. The request fails if the packed size is greater than 100 percent, which means the policies and tags exceeded the allowed space.

Type: Integer

Valid Range: Minimum value of 0.

SourceIdentity

The value in the `SourceIdentity` attribute in the SAML assertion.

You can require users to set a source identity value when they assume a role. You do this by using the `sts:SourceIdentity` condition key in a role trust policy. That way, actions that are taken with the role are associated with that user. After the source identity is set, the value cannot be changed. It is present in the request for all actions that are taken by the role and persists across chained role sessions. You can configure your SAML identity provider to use an attribute associated with your users, like user name or email, as the source identity when calling `AssumeRoleWithSAML`. You do this by adding an attribute to the SAML assertion. For more information about using source identity, see Monitor and control actions taken with assumed roles in the *IAM User Guide*.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

Subject

The value of the `NameID` element in the `Subject` element of the SAML assertion.

Type: String

SubjectType

The format of the name ID, as defined by the `Format` attribute in the `NameID` element of the SAML assertion. Typical examples of the format are `transient` or `persistent`.

If the format includes the prefix `urn:oasis:names:tc:SAML:2.0:nameid-format`, that prefix is removed. For example, `urn:oasis:names:tc:SAML:2.0:nameid-format:transient` is returned as `transient`. If the format includes any other prefix, the format is returned with no modifications.

Type: String

Errors

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

ExpiredToken

The web identity token that was passed is expired or is not valid. Get a new identity token from the identity provider and then retry the request.

HTTP Status Code: 400

IDPRejectedClaim

The identity provider (IdP) reported that authentication failed. This might be because the claim is invalid.

If this error is returned for the `AssumeRoleWithWebIdentity` operation, it can also mean that the claim has expired or has been explicitly revoked.

HTTP Status Code: 403

InvalidIdentityToken

The web identity token that was passed could not be validated by AWS. Get a new identity token from the identity provider and then retry the request.

HTTP Status Code: 400

MalformedPolicyDocument

The request was rejected because the policy document was malformed. The error message describes the specific error.

HTTP Status Code: 400

PackedPolicyTooLarge

The request was rejected because the total packed size of the session policies and session tags combined was too large. An AWS conversion compresses the session policy document, session policy ARNs, and session tags into a packed binary format that has a separate limit. The error message indicates by percentage how close the policies and tags are to the upper size limit. For more information, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You could receive this error even though you meet other defined session policy and session tag limits. For more information, see IAM and AWS STS Entity Character Limits in the *IAM User Guide*.

HTTP Status Code: 400

RegionDisabled

AWS STS is not activated in the requested region for the account that is being asked to generate credentials. The account administrator must use the IAM console to activate AWS STS in that region. For more information, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

HTTP Status Code: 403

Examples

Example

This example requires that the `TestSaml` role and the `SAML-test` SAML identity provider are configured in IAM. Additionally, this example requires an encoded SAML assertion that includes all

of the necessary information. For more information about SAML assertions, see Configuring SAML Assertions for the Authentication Response in the *IAM User Guide*.

Sample Request

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=AssumeRoleWithSAML
&RoleArn=arn:aws:iam::123456789012:role/TestSaml
&PrincipalArn=arn:aws:iam::123456789012:saml-provider/SAML-test
&SAMLAssertion=VERYLONGENCODEDASSERTIONEXAMPLExzYW1sOkF1ZGllbmNl
PmJsYW5rPC9zYW1sOkF1ZGllbmNlPjwvc2FtbDpBdWRpZW5jZVJlc3RyaWN0aW9u
Pjwvc2FtbDpDb25kaXRpb25zPjxzYW1sOlN1YmplY3Q+PHNhbWw6TmFtZUlEIEZv
cm1hdD0idXJuOm9hc2lzOm5hbWVzOnRjOlNBTUw6Mi4wOm5hbWVpZC1mb3JtYXQ6
dHJhbnNpZW50Ij5TYW1sRXhhbXBsZTwvc2FtbDpOYW1lSUQ+PHNhbWw6U3ViamVj
dENvbmZpcm1hdGlvbiBNZXRob2Q9InVybjpvYXNpczpuYW1lczp0YzpTQU1MOjIu
MDpjbTpiZWFyZXIiPjxzYW1sOlN1YmplY3RDb25maXJtYXRpb25EYXRhIE5vdE9u
T3JBZnRlcj0iMjAxOS0xMS0wMVQyMDoyNTowNS4xNDVaIiBSZWNpcGllbnQ9Imh0
dHBzOi8vc2lnbmluLmF3cy5hbWF6b24uY29tL3NhbWwiLz48L3NhbWw6U3ViamVj
dENvbmZpcm1hdGlvbj48L3NhbWw6U3ViamVjdD48c2FtbDpBdXRoblN0YXRlbWVu
dCBBdXRoPD94bWwgdmpSZXNwb25zZT4=
&AUTHPARAMS
```

Sample Response

```
<AssumeRoleWithSAMLResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
    <AssumeRoleWithSAMLResult>
        <Issuer> https://integ.example.com/idp/shibboleth</Issuer>
        <AssumedRoleUser>
            <Arn>arn:aws:sts::123456789012:assumed-role/TestSaml</Arn>
            <AssumedRoleId>ARO456EXAMPLE789:TestSaml</AssumedRoleId>
        </AssumedRoleUser>
        <Credentials>
            <AccessKeyId>ASIAV3ZUEFP6EXAMPLE</AccessKeyId>
            <SecretAccessKey>8P+SQvWIuLnKhh8d++jpw0nNmQRBZvNEXAMPLEKEY</
SecretAccessKey>
            <SessionToken> IQoJb3JpZ2luX2VjEOz////////////////////
wEXAMPLEtMSJHMEUCIDoKK3JH9uG
                QE1z0sINr5M4jk+Na8KHDcCYRVjJCZEvOAiEA3OvJGtw1EcViOleS2vhs8VdCKFJQWP
                QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
                9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
                +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA== </
SessionToken>
```

```
            <Expiration>2019-11-01T20:26:47Z</Expiration>
        </Credentials>
        <Audience>https://signin.aws.amazon.com/saml</Audience>
        <SubjectType>transient</SubjectType>
        <PackedPolicySize>6</PackedPolicySize>
        <NameQualifier>SbdGOnUkh1i4+EXAMPLExL/jEvs=</NameQualifier>
        <SourceIdentity>SourceIdentityValue</SourceIdentity>
        <Subject>SamlExample</Subject>
    </AssumeRoleWithSAMLResult>
    <ResponseMetadata>
        <RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
    </ResponseMetadata>
</AssumeRoleWithSAMLResponse>
```

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

AssumeRoleWithWebIdentity

Returns a set of temporary security credentials for users who have been authenticated in a mobile or web application with a web identity provider. Example providers include the OAuth 2.0 providers Login with Amazon and Facebook, or any OpenID Connect-compatible identity provider such as Google or Amazon Cognito federated identities.

> **Note**
>
> For mobile applications, we recommend that you use Amazon Cognito. You can use Amazon Cognito with the AWS SDK for iOS Developer Guide and the AWS SDK for Android Developer Guide to uniquely identify a user. You can also supply the user with a consistent identity throughout the lifetime of an application.
> To learn more about Amazon Cognito, see Amazon Cognito identity pools in *Amazon Cognito Developer Guide*.

Calling `AssumeRoleWithWebIdentity` does not require the use of AWS security credentials. Therefore, you can distribute an application (for example, on mobile devices) that requests temporary security credentials without including long-term AWS credentials in the application. You also don't need to deploy server-based proxy services that use long-term AWS credentials. Instead, the identity of the caller is validated by using a token from the web identity provider. For a comparison of `AssumeRoleWithWebIdentity` with the other API operations that produce temporary credentials, see Requesting Temporary Security Credentials and Comparing the AWS STS API operations in the *IAM User Guide*.

The temporary security credentials returned by this API consist of an access key ID, a secret access key, and a security token. Applications can use these temporary security credentials to sign calls to AWS service API operations.

Session Duration

By default, the temporary security credentials created by `AssumeRoleWithWebIdentity` last for one hour. However, you can use the optional `DurationSeconds` parameter to specify the duration of your session. You can provide a value from 900 seconds (15 minutes) up to the maximum session duration setting for the role. This setting can have a value from 1 hour to 12 hours. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*. The maximum session duration limit applies when you use the

`AssumeRole*` API operations or the `assume-role*` CLI commands. However the limit does not apply when you use those operations to create a console URL. For more information, see Using IAM Roles in the *IAM User Guide*.

Permissions

The temporary security credentials created by `AssumeRoleWithWebIdentity` can be used to make API calls to any AWS service with the following exception: you cannot call the AWS STS `GetFederationToken` or `GetSessionToken` API operations.

(Optional) You can pass inline or managed session policies to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies. The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

Tags

(Optional) You can configure your IdP to pass attributes into your web identity token as session tags. Each session tag consists of a key name and an associated value. For more information about session tags, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You can pass up to 50 session tags. The plaintext session tag keys can’t exceed 128 characters and the values can’t exceed 256 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

You can pass a session tag with the same key as a tag that is attached to the role. When you do, the session tag overrides the role tag with the same key.

An administrator must grant you the permissions necessary to pass session tags. The administrator can also create granular permissions to allow you to pass only specific session tags. For more information, see Tutorial: Using Tags for Attribute-Based Access Control in the *IAM User Guide*.

You can set the session tags as transitive. Transitive tags persist during role chaining. For more information, see Chaining Roles with Session Tags in the *IAM User Guide*.

Identities

Before your application can call `AssumeRoleWithWebIdentity`, you must have an identity token from a supported identity provider and create a role that the application can assume. The role that your application assumes must trust the identity provider that is associated with the identity token. In other words, the identity provider must be specified in the role's trust policy.

> **Important**
>
> Calling `AssumeRoleWithWebIdentity` can result in an entry in your AWS CloudTrail logs. The entry includes the Subject of the provided web identity token. We recommend that you avoid using any personally identifiable information (PII) in this field. For example, you could instead use a GUID or a pairwise identifier, as suggested in the OIDC specification.

For more information about how to use web identity federation and the `AssumeRoleWithWebIdentity` API, see the following resources:

- Using Web Identity Federation API Operations for Mobile Apps and Federation Through a Web-based Identity Provider.
- Web Identity Federation Playground. Walk through the process of authenticating through Login with Amazon, Facebook, or Google, getting temporary security credentials, and then using those credentials to make a request to AWS.
- AWS SDK for iOS Developer Guide and AWS SDK for Android Developer Guide. These toolkits contain sample apps that show how to invoke the identity providers. The toolkits then show how to use the information from these providers to get and use temporary security credentials.
- Web Identity Federation with Mobile Applications. This article discusses web identity federation and shows an example of how to use web identity federation to get access to content in Amazon S3.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

DurationSeconds

The duration, in seconds, of the role session. The value can range from 900 seconds (15 minutes) up to the maximum session duration setting for the role. This setting can have a value from 1 hour to 12 hours. If you specify a value higher than this setting, the operation fails. For example, if you specify a session duration of 12 hours, but your administrator set the maximum session duration to 6 hours, your operation fails. To learn how to view the maximum value for your role, see View the Maximum Session Duration Setting for a Role in the *IAM User Guide*.

By default, the value is set to `3600` seconds.

> **Note**
>
> The `DurationSeconds` parameter is separate from the duration of a console session that you might request using the returned credentials. The request to the federation endpoint for a console sign-in token takes a `SessionDuration` parameter that specifies the maximum length of the console session. For more information, see Creating a URL that Enables Federated Users to Access the AWS Management Console in the *IAM User Guide*.

Type: Integer

Valid Range: Minimum value of 900. Maximum value of 43200.

Required: No

Policy

An IAM policy in JSON format that you want to use as an inline session policy.

This parameter is optional. Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. The JSON policy characters can be any ASCII character from the space character to the end of the valid character list (\u0020 through \u00FF). It can also include the tab (\u0009), linefeed (\u000A), and carriage return (\u000D) characters.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]+`

Required: No

PolicyArns.member.N

The Amazon Resource Names (ARNs) of the IAM managed policies that you want to use as managed session policies. The policies must exist in the same account as the role.

This parameter is optional. You can provide up to 10 managed policy ARNs. However, the plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. For more information about ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the AWS General Reference.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Passing policies to this operation returns new temporary credentials. The resulting session's permissions are the intersection of the role's identity-based policy and the session policies. You can use the role's temporary credentials in subsequent AWS API calls to access resources in the account that owns the role. You cannot use session policies to grant more permissions than those allowed by the identity-based policy of the role that is being assumed. For more information, see Session Policies in the *IAM User Guide*.

Type: Array of PolicyDescriptorType objects

Required: No

ProviderId

The fully qualified host component of the domain name of the OAuth 2.0 identity provider. Do not specify this value for an OpenID Connect identity provider.

Currently `www.amazon.com` and `graph.facebook.com` are the only supported identity providers for OAuth 2.0 access tokens. Do not include URL schemes and port numbers.

Do not specify this value for OpenID Connect ID tokens.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 2048.

Required: No

RoleArn

The Amazon Resource Name (ARN) of the role that the caller is assuming.

> **Note**
>
> Additional considerations apply to Amazon Cognito identity pools that assume cross-account IAM roles. The trust policies of these roles must accept the `cognito-identity.amazonaws.com` service principal and must contain the `cognito-identity.amazonaws.com:aud` condition key to restrict role assumption to users from your intended identity pools. A policy that trusts Amazon Cognito identity pools without this condition creates a risk that a user from an unintended identity pool can assume the role. For more information, see Trust policies for IAM roles in Basic (Classic) authentication in the *Amazon Cognito Developer Guide*.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

RoleSessionName

An identifier for the assumed role session. Typically, you pass the name or identifier that is associated with the user who is using your application. That way, the temporary security credentials that your application will use are associated with that user. This session name is included as part of the ARN and assumed role ID in the `AssumedRoleUser` response element.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

Required: Yes

WebIdentityToken

The OAuth 2.0 access token or OpenID Connect ID token that is provided by the identity provider. Your application must get this token by authenticating the user who is using your application with a web identity provider before the application makes an `AssumeRoleWithWebIdentity` call. Only tokens with RSA algorithms (RS256) are supported.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 20000.

Required: Yes

Response Elements

The following elements are returned by the service.

AssumedRoleUser

The Amazon Resource Name (ARN) and the assumed role ID, which are identifiers that you can use to refer to the resulting temporary security credentials. For example, you can reference these credentials as a principal in a resource-based policy by using the ARN or assumed role ID. The ARN and ID include the `RoleSessionName` that you specified when you called `AssumeRole`.

Type: AssumedRoleUser object

Audience

The intended audience (also known as client ID) of the web identity token. This is traditionally the client identifier issued to the application that requested the web identity token.

Type: String

Credentials

The temporary security credentials, which include an access key ID, a secret access key, and a security token.

> **Note**
>
> The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size.

Type: Credentials object

PackedPolicySize

A percentage value that indicates the packed size of the session policies and session tags combined passed in the request. The request fails if the packed size is greater than 100 percent, which means the policies and tags exceeded the allowed space.

Type: Integer

Valid Range: Minimum value of 0.

Provider

The issuing authority of the web identity token presented. For OpenID Connect ID tokens, this contains the value of the `iss` field. For OAuth 2.0 access tokens, this contains the value of the `ProviderId` parameter that was passed in the `AssumeRoleWithWebIdentity` request.

Type: String

SourceIdentity

The value of the source identity that is returned in the JSON web token (JWT) from the identity provider.

You can require users to set a source identity value when they assume a role. You do this by using the `sts:SourceIdentity` condition key in a role trust policy. That way, actions that are taken with the role are associated with that user. After the source identity is set, the value cannot be changed. It is present in the request for all actions that are taken by the role and persists across chained role sessions. You can configure your identity provider to use an attribute associated with your users, like user name or email, as the source identity when calling `AssumeRoleWithWebIdentity`. You do this by adding a claim to the JSON web token. To learn more about OIDC tokens and claims, see Using Tokens with User Pools in the *Amazon Cognito Developer Guide*. For more information about using source identity, see Monitor and control actions taken with assumed roles in the *IAM User Guide*.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 64.

Pattern: `[\w+=,.@-]*`

SubjectFromWebIdentityToken

The unique user identifier that is returned by the identity provider. This identifier is associated with the `WebIdentityToken` that was submitted with the `AssumeRoleWithWebIdentity` call. The identifier is typically unique to the user and the application that acquired the `WebIdentityToken` (pairwise identifier). For OpenID Connect ID tokens, this field contains the value returned by the identity provider as the token's `sub` (Subject) claim.

Type: String

Length Constraints: Minimum length of 6. Maximum length of 255.

Errors

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

ExpiredToken

The web identity token that was passed is expired or is not valid. Get a new identity token from the identity provider and then retry the request.

HTTP Status Code: 400

IDPCommunicationError

The request could not be fulfilled because the identity provider (IDP) that was asked to verify the incoming identity token could not be reached. This is often a transient error caused by network conditions. Retry the request a limited number of times so that you don't exceed the request rate. If the error persists, the identity provider might be down or not responding.

HTTP Status Code: 400

IDPRejectedClaim

The identity provider (IdP) reported that authentication failed. This might be because the claim is invalid.

If this error is returned for the `AssumeRoleWithWebIdentity` operation, it can also mean that the claim has expired or has been explicitly revoked.

HTTP Status Code: 403

InvalidIdentityToken

The web identity token that was passed could not be validated by AWS. Get a new identity token from the identity provider and then retry the request.

HTTP Status Code: 400

MalformedPolicyDocument

The request was rejected because the policy document was malformed. The error message describes the specific error.

HTTP Status Code: 400

PackedPolicyTooLarge

The request was rejected because the total packed size of the session policies and session tags combined was too large. An AWS conversion compresses the session policy document, session policy ARNs, and session tags into a packed binary format that has a separate limit. The error message indicates by percentage how close the policies and tags are to the upper size limit. For more information, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You could receive this error even though you meet other defined session policy and session tag limits. For more information, see IAM and AWS STS Entity Character Limits in the *IAM User Guide*.

HTTP Status Code: 400

RegionDisabled

AWS STS is not activated in the requested region for the account that is being asked to generate credentials. The account administrator must use the IAM console to activate AWS STS in that region. For more information, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

HTTP Status Code: 403

Examples

Example

This example illustrates one usage of AssumeRoleWithWebIdentity.

Sample Request

```
https://sts.amazonaws.com/
?Action=AssumeRoleWithWebIdentity
&DurationSeconds=3600
&PolicyArns.member.1.arn=arn:aws:iam::123456789012:policy/webidentitydemopolicy1
&PolicyArns.member.2.arn=arn:aws:iam::123456789012:policy/webidentitydemopolicy2
&ProviderId=www.amazon.com
&RoleSessionName=app1
&RoleArn=arn:aws:iam::123456789012:role/FederatedWebIdentityRole
&WebIdentityToken=Atza%7CIQEBLjAsAhRFiXuWpUXuRvQ9PZL3GMFcYevydwIUFAHZwXZXX
```

```
XXXXXXJnrulxKDHwy87oGKPznh0D6bEQZTSCzyoCtL_8S07pLpr0zMbn6w1lfVZKNTBdDansFB
mtGnIsIapjI6xKR02Yc_2bQ8LZbUXSGm6Ry6_BG7PrtLZtj_dfCTj92xNGed-CrKqjG7nPBjNI
L016GGvuS5gSvPRUxWES3VYfm1wl7WTI7jn-Pcb6M-buCgHhFOzTQxod27L9CqnOLio7N3gZAG
psp6n1-AJBOCJckcyXe2c6uD0srOJeZlKUm2eTDVMf8IehDVI0r1QOnTV6KzzAI3OY87Vd_cVMQ
&Version=2011-06-15
```

Sample Response

```
<AssumeRoleWithWebIdentityResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <AssumeRoleWithWebIdentityResult>
    <SubjectFromWebIdentityToken>amzn1.account.AF6RHO7KZU5XRVQJGXK6HB56KR2A</
SubjectFromWebIdentityToken>
    <Audience>client.5498841531868486423.1548@apps.example.com</Audience>
    <AssumedRoleUser>
      <Arn>arn:aws:sts::123456789012:assumed-role/FederatedWebIdentityRole/app1</Arn>
      <AssumedRoleId>AROACLKWSDQRAOEXAMPLE:app1</AssumedRoleId>
    </AssumedRoleUser>
    <Credentials>
      <SessionToken>AQoDYXdzEE0a8ANXXXXXXXXNO1ewxE5TijQyp+IEXAMPLE</SessionToken>
      <SecretAccessKey>wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY</SecretAccessKey>
      <Expiration>2014-10-24T23:00:23Z</Expiration>
      <AccessKeyId>ASgeIAIOSFODNN7EXAMPLE</AccessKeyId>
    </Credentials>
    <SourceIdentity>SourceIdentityValue</SourceIdentity>
    <Provider>www.amazon.com</Provider>
  </AssumeRoleWithWebIdentityResult>
  <ResponseMetadata>
    <RequestId>ad4156e9-bce1-11e2-82e6-6b6efEXAMPLE</RequestId>
  </ResponseMetadata>
</AssumeRoleWithWebIdentityResponse>
```

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

DecodeAuthorizationMessage

Decodes additional information about the authorization status of a request from an encoded message returned in response to an AWS request.

For example, if a user is not authorized to perform an operation that he or she has requested, the request returns a `Client.UnauthorizedOperation` response (an HTTP 403 response). Some AWS operations additionally return an encoded message that can provide details about this authorization failure.

> **Note**
>
> Only certain AWS operations return an encoded authorization message. The documentation for an individual operation indicates whether that operation returns an encoded message in addition to returning an HTTP code.

The message is encoded because the details of the authorization status can contain privileged information that the user who requested the operation should not see. To decode an authorization status message, a user must be granted permissions through an IAM policy to request the `DecodeAuthorizationMessage` (`sts:DecodeAuthorizationMessage`) action.

The decoded message includes the following type of information:

- Whether the request was denied due to an explicit deny or due to the absence of an explicit allow. For more information, see Determining Whether a Request is Allowed or Denied in the *IAM User Guide*.
- The principal who made the request.
- The requested action.
- The requested resource.
- The values of condition keys in the context of the user's request.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

EncodedMessage

The encoded message that was returned with the response.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 10240.

Required: Yes

Response Elements

The following element is returned by the service.

DecodedMessage

The API returns a response with the decoded message.

Type: String

Errors

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

InvalidAuthorizationMessage

The error returned if the message passed to `DecodeAuthorizationMessage` was invalid. This can happen if the token contains invalid characters, such as linebreaks.

HTTP Status Code: 400

Examples

Example

The following code provides an example of a `DecodeAuthorizationMessage` policy:

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

```
    {
      "Sid": "decodepolicy",
      "Effect": "Allow",
      "Action": "sts:DecodeAuthorizationMessage",
      "Resource": "*"
    }
  ]
}
```

Example

This example illustrates one usage of DecodeAuthorizationMessage.

Sample Request

```
POST https://sts.amazonaws.com / HTTP/1.1
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: sts.amazonaws.com
Content-Length: 1148
Expect: 100-continue
Connection: Keep-Alive
Action=DecodeAuthorizationMessage
&EncodedMessage=<encoded-message>
&Version=2011-06-15
&AUTHPARAMS
```

Sample Response

```
<?xml version="1.0" encoding="UTF-8"?>
<DecodeAuthorizationMessageResponse xmlns="http://sts.amazonaws.com/doc/2011-06-15/">
    <requestId>6624a9ca-cd25-4f50-b2a5-7ba65bf07453</requestId>
    <DecodedMessage>
    {
      "allowed": "false",
      "explicitDeny": "false",
      "matchedStatements": "",
      "failures": "",
      "context": {
        "principal": {
          "id": "AIDACKCEVSQ6C2EXAMPLE",
          "name": "Bob",
          "arn": "arn:aws:iam::123456789012:user/Bob"
        },
```

```
        "action": "ec2:StopInstances",
        "resource": "arn:aws:ec2:us-east-1:123456789012:instance/i-dd01c9bd",
        "conditions": [
          {
            "item": {
              "key": "ec2:Tenancy",
              "values": ["default"]
             },
          {
            "item": {
              "key": "ec2:ResourceTag/elasticbeanstalk:environment-name",
              "values": ["Default-Environment"]
            }
          },
          (Additional items ...)
        ]
      }
    }
    </DecodedMessage>
</DecodeAuthorizationMessageResponse>
```

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

GetAccessKeyInfo

Returns the account identifier for the specified access key ID.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). For more information about access keys, see Managing Access Keys for IAM Users in the *IAM User Guide*.

When you pass an access key ID to this operation, it returns the ID of the AWS account to which the keys belong. Access key IDs beginning with `AKIA` are long-term credentials for an IAM user or the AWS account root user. Access key IDs beginning with `ASIA` are temporary credentials that are created using AWS STS operations. If the account in the response belongs to you, you can sign in as the root user and review your root user access keys. Then, you can pull a credentials report to learn which IAM user owns the keys. To learn who requested the temporary credentials for an `ASIA` access key, view the AWS STS events in your CloudTrail logs in the *IAM User Guide*.

This operation does not indicate the state of the access key. The key might be active, inactive, or deleted. Active keys might not have permissions to perform an operation. Providing a deleted access key might return an error that the key doesn't exist.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

AccessKeyId

The identifier of an access key.

This parameter allows (through its regex pattern) a string of characters that can consist of any upper- or lowercase letter or digit.

Type: String

Length Constraints: Minimum length of 16. Maximum length of 128.

Pattern: `[\w]*`

Required: Yes

Response Elements

The following element is returned by the service.

Account

The number used to identify the AWS account.

Type: String

Errors

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

Examples

Example

This example illustrates one usage of GetAccessKeyInfo.

Sample Request

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetAccessKeyInfo
&AccessKeyId=AKIAI44QH8DHBEXAMPLE
&AUTHPARAMS
```

Sample Response

```
<GetAccessKeyInfoResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetAccessKeyInfoResult>
    <Account>111122223333</Account>
  </GetAccessKeyInfoResult>
 <ResponseMetadata>
    <RequestId>7a62c49f-347e-4fc4-9331-6e8eEXAMPLE</RequestId>
 </ResponseMetadata>
</GetAccessKeyInfoResponse>
```

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

GetCallerIdentity

Returns details about the IAM user or role whose credentials are used to call the operation.

> **Note**
>
> No permissions are required to perform this operation. If an administrator attaches a policy to your identity that explicitly denies access to the `sts:GetCallerIdentity` action, you can still perform this operation. Permissions are not required because the same information is returned when access is denied. To view an example response, see I Am Not Authorized to Perform: iam:DeleteVirtualMFADevice in the *IAM User Guide*.

Response Elements

The following elements are returned by the service.

Account

The AWS account ID number of the account that owns or contains the calling entity.

Type: String

Arn

The AWS ARN associated with the calling entity.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

UserId

The unique identifier of the calling entity. The exact value depends on the type of entity that is making the call. The values returned are those listed in the **aws:userid** column in the Principal table found on the **Policy Variables** reference page in the *IAM User Guide*.

Type: String

Errors

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

Examples

Example 1 - Called by an IAM user

This example shows a request and response made with the credentials for a user named Alice in the AWS account 123456789012.

Sample Request

```
POST / HTTP/1.1
Host: sts.amazonaws.com
Accept-Encoding: identity
Content-Length: 32
Content-Type: application/x-www-form-urlencoded
Authorization: AWS4-HMAC-SHA256 Credential=AKIAI44QH8DHBEXAMPLE/20160126/us-east-1/sts/
aws4_request,
        SignedHeaders=host;user-agent;x-amz-date,
        Signature=1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef
X-Amz-Date: 20160126T215751Z
User-Agent: aws-cli/1.10.0 Python/2.7.3 Linux/3.13.0-76-generic botocore/1.3.22

Action=GetCallerIdentity&Version=2011-06-15
```

Sample Response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: text/xml
Content-Length: 357
Date: Tue, 26 Jan 2016 21:57:47 GMT

<GetCallerIdentityResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetCallerIdentityResult>
  <Arn>arn:aws:iam::123456789012:user/Alice</Arn>
  <UserId>AIDACKCEVSQ6C2EXAMPLE</UserId>
  <Account>123456789012</Account>
  </GetCallerIdentityResult>
  <ResponseMetadata>
```

```
    <RequestId>01234567-89ab-cdef-0123-456789abcdef</RequestId>
  </ResponseMetadata>
</GetCallerIdentityResponse>
```

Example 2 - Called by user created with AssumeRole

This example shows a request and response made with temporary credentials created by `AssumeRole`. The name of the assumed role is `my-role-name`, and the `RoleSessionName` is set to `my-role-session-name`.

Sample Request

```
POST / HTTP/1.1
Host: sts.amazonaws.com
Accept-Encoding: identity
Content-Length: 43
X-Amz-Date: 20160301T213302Z
User-Agent: aws-cli/1.10.0 Python/2.7.3 Linux/3.13.0-79-generic botocore/1.3.22
X-Amz-Security-Token:<REDACTED>
Content-Type: application/x-www-form-urlencoded
Authorization: AWS4-HMAC-SHA256 Credential=AKIAI44QH8DHBEXAMPLE/20160301/us-east-1/sts/
aws4_request,
        SignedHeaders=host;user-agent;x-amz-date;x-amz-security-token,
        Signature=1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef

Action=GetCallerIdentity&Version=2011-06-15
```

Sample Response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: text/xml
Content-Length: 438
Date: Tue, 01 Mar 2016 21:32:59 GMT

<GetCallerIdentityResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetCallerIdentityResult>
    <Arn>arn:aws:sts::123456789012:assumed-role/my-role-name/my-role-session-name</Arn>
    <UserId>ARO123EXAMPLE123:my-role-session-name</UserId>
    <Account>123456789012</Account>
  </GetCallerIdentityResult>
```

```
  <ResponseMetadata>
    <RequestId>01234567-89ab-cdef-0123-456789abcdef</RequestId>
  </ResponseMetadata>
</GetCallerIdentityResponse>
```

Example 3 - Called by user created with GetFederationToken

This example shows a request and response made with temporary credentials created by using `GetFederationToken`. The `Name` parameter is set to `my-federated-user-name`.

Sample Request

```
POST / HTTP/1.1
Host: sts.amazonaws.com
Accept-Encoding: identity
Content-Length: 43
X-Amz-Date: 20160301T215108Z
User-Agent: aws-cli/1.10.0 Python/2.7.3 Linux/3.13.0-79-generic botocore/1.3.22
X-Amz-Security-Token:<REDACTED>
Content-Type: application/x-www-form-urlencoded
Authorization: AWS4-HMAC-SHA256 Credential=AKIAI44QH8DHBEXAMPLE/20160301/us-east-1/sts/
aws4_request,
        SignedHeaders=host;user-agent;x-amz-date;x-amz-security-token,
        Signature=1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef

Action=GetCallerIdentity&Version=2011-06-15
```

Sample Response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 01234567-89ab-cdef-0123-456789abcdef
Content-Type: text/xml
Content-Length: 437
Date: Tue, 01 Mar 2016 21:51:06 GMT

<GetCallerIdentityResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetCallerIdentityResult>
    <Arn>arn:aws:sts::123456789012:federated-user/my-federated-user-name</Arn>
    <UserId>123456789012:my-federated-user-name</UserId>
    <Account>123456789012</Account>
  </GetCallerIdentityResult>
  <ResponseMetadata>
```

```
    <RequestId>01234567-89ab-cdef-0123-456789abcdef</RequestId>
  </ResponseMetadata>
</GetCallerIdentityResponse>
```

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

GetFederationToken

Returns a set of temporary security credentials (consisting of an access key ID, a secret access key, and a security token) for a user. A typical use is in a proxy application that gets temporary security credentials on behalf of distributed applications inside a corporate network.

You must call the `GetFederationToken` operation using the long-term security credentials of an IAM user. As a result, this call is appropriate in contexts where those credentials can be safeguarded, usually in a server-based application. For a comparison of `GetFederationToken` with the other API operations that produce temporary credentials, see Requesting Temporary Security Credentials and Comparing the AWS STS API operations in the *IAM User Guide*.

Although it is possible to call `GetFederationToken` using the security credentials of an AWS account root user rather than an IAM user that you create for the purpose of a proxy application, we do not recommend it. For more information, see Safeguard your root user credentials and don't use them for everyday tasks in the *IAM User Guide*.

> **Note**
>
> You can create a mobile-based or browser-based app that can authenticate users using a web identity provider like Login with Amazon, Facebook, Google, or an OpenID Connect-compatible identity provider. In this case, we recommend that you use Amazon Cognito or `AssumeRoleWithWebIdentity`. For more information, see Federation Through a Web-based Identity Provider in the *IAM User Guide*.

Session duration

The temporary credentials are valid for the specified duration, from 900 seconds (15 minutes) up to a maximum of 129,600 seconds (36 hours). The default session duration is 43,200 seconds (12 hours). Temporary credentials obtained by using the root user credentials have a maximum duration of 3,600 seconds (1 hour).

Permissions

You can use the temporary credentials created by `GetFederationToken` in any AWS service with the following exceptions:

- You cannot call any IAM operations using the AWS CLI or the AWS API. This limitation does not apply to console sessions.

- You cannot call any AWS STS operations except `GetCallerIdentity`.

You can use temporary credentials for single sign-on (SSO) to the console.

You must pass an inline or managed session policy to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies. The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters.

Though the session policy parameters are optional, if you do not pass a policy, then the resulting federated user session has no permissions. When you pass session policies, the session permissions are the intersection of the IAM user policies and the session policies that you pass. This gives you a way to further restrict the permissions for a federated user. You cannot use session policies to grant more permissions than those that are defined in the permissions policy of the IAM user. For more information, see Session Policies in the *IAM User Guide*. For information about using `GetFederationToken` to create temporary security credentials, see GetFederationToken—Federation Through a Custom Identity Broker.

You can use the credentials to access a resource that has a resource-based policy. If that policy specifically references the federated user session in the `Principal` element of the policy, the session has the permissions allowed by the policy. These permissions are granted in addition to the permissions granted by the session policies.

Tags

(Optional) You can pass tag key-value pairs to your session. These are called session tags. For more information about session tags, see Passing Session Tags in AWS STS in the *IAM User Guide*.

> **Note**
>
> You can create a mobile-based or browser-based app that can authenticate users using a web identity provider like Login with Amazon, Facebook, Google, or an OpenID Connect-compatible identity provider. In this case, we recommend that you use Amazon Cognito or `AssumeRoleWithWebIdentity`. For more information, see Federation Through a Web-based Identity Provider in the *IAM User Guide*.

An administrator must grant you the permissions necessary to pass session tags. The administrator can also create granular permissions to allow you to pass only specific session tags. For more information, see Tutorial: Using Tags for Attribute-Based Access Control in the *IAM User Guide*.

Tag key–value pairs are not case sensitive, but case is preserved. This means that you cannot have separate `Department` and `department` tag keys. Assume that the user that you are federating has the `Department=Marketing` tag and you pass the `department=engineering` session tag. `Department` and `department` are not saved as separate tags, and the session tag passed in the request takes precedence over the user tag.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

DurationSeconds

The duration, in seconds, that the session should last. Acceptable durations for federation sessions range from 900 seconds (15 minutes) to 129,600 seconds (36 hours), with 43,200 seconds (12 hours) as the default. Sessions obtained using root user credentials are restricted to a maximum of 3,600 seconds (one hour). If the specified duration is longer than one hour, the session obtained by using root user credentials defaults to one hour.

Type: Integer

Valid Range: Minimum value of 900. Maximum value of 129600.

Required: No

Name

The name of the federated user. The name is used as an identifier for the temporary security credentials (such as Bob). For example, you can reference the federated user name in a resource-based policy, such as in an Amazon S3 bucket policy.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@-

Type: String

Length Constraints: Minimum length of 2. Maximum length of 32.

Pattern: `[\w+=,.@-]*`

Required: Yes

Policy

An IAM policy in JSON format that you want to use as an inline session policy.

You must pass an inline or managed session policy to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies.

This parameter is optional. However, if you do not pass any session policies, then the resulting federated user session has no permissions.

When you pass session policies, the session permissions are the intersection of the IAM user policies and the session policies that you pass. This gives you a way to further restrict the permissions for a federated user. You cannot use session policies to grant more permissions than those that are defined in the permissions policy of the IAM user. For more information, see Session Policies in the *IAM User Guide*.

The resulting credentials can be used to access a resource that has a resource-based policy. If that policy specifically references the federated user session in the `Principal` element of the policy, the session has the permissions allowed by the policy. These permissions are granted in addition to the permissions that are granted by the session policies.

The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. The JSON policy characters can be any ASCII character from the space character to the end of the valid character list (\u0020 through \u00FF). It can also include the tab (\u0009), linefeed (\u000A), and carriage return (\u000D) characters.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u00FF]+`

Required: No

PolicyArns.member.N

The Amazon Resource Names (ARNs) of the IAM managed policies that you want to use as a managed session policy. The policies must exist in the same account as the IAM user that is requesting federated access.

You must pass an inline or managed session policy to this operation. You can pass a single JSON policy document to use as an inline session policy. You can also specify up to 10 managed policy Amazon Resource Names (ARNs) to use as managed session policies. The plaintext that you use for both inline and managed session policies can't exceed 2,048 characters. You can provide up to 10 managed policy ARNs. For more information about ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the AWS General Reference.

This parameter is optional. However, if you do not pass any session policies, then the resulting federated user session has no permissions.

When you pass session policies, the session permissions are the intersection of the IAM user policies and the session policies that you pass. This gives you a way to further restrict the permissions for a federated user. You cannot use session policies to grant more permissions than those that are defined in the permissions policy of the IAM user. For more information, see Session Policies in the *IAM User Guide*.

The resulting credentials can be used to access a resource that has a resource-based policy. If that policy specifically references the federated user session in the `Principal` element of the policy, the session has the permissions allowed by the policy. These permissions are granted in addition to the permissions that are granted by the session policies.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

Type: Array of PolicyDescriptorType objects

Required: No

Tags.member.N

A list of session tags. Each session tag consists of a key name and an associated value. For more information about session tags, see Passing Session Tags in AWS STS in the *IAM User Guide*.

This parameter is optional. You can pass up to 50 session tags. The plaintext session tag keys can't exceed 128 characters and the values can't exceed 256 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

> **Note**
>
> An AWS conversion compresses the passed inline session policy, managed policy ARNs, and session tags into a packed binary format that has a separate limit. Your request can fail for this limit even if your plaintext meets the other requirements. The `PackedPolicySize` response element indicates by percentage how close the policies and tags for your request are to the upper size limit.

You can pass a session tag with the same key as a tag that is already attached to the user you are federating. When you do, session tags override a user tag with the same key.

Tag key–value pairs are not case sensitive, but case is preserved. This means that you cannot have separate `Department` and `department` tag keys. Assume that the role has the `Department=Marketing` tag and you pass the `department=engineering` session tag. `Department` and `department` are not saved as separate tags, and the session tag passed in the request takes precedence over the role tag.

Type: Array of Tag objects

Array Members: Maximum number of 50 items.

Required: No

Response Elements

The following elements are returned by the service.

Credentials

The temporary security credentials, which include an access key ID, a secret access key, and a security (or session) token.

> **Note**
>
> The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size.

Type: Credentials object

FederatedUser

Identifiers for the federated user associated with the credentials (such as `arn:aws:sts::123456789012:federated-user/Bob` or `123456789012:Bob`). You can use the federated user's ARN in your resource-based policies, such as an Amazon S3 bucket policy.

Type: FederatedUser object

PackedPolicySize

A percentage value that indicates the packed size of the session policies and session tags combined passed in the request. The request fails if the packed size is greater than 100 percent, which means the policies and tags exceeded the allowed space.

Type: Integer

Valid Range: Minimum value of 0.

Errors

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

MalformedPolicyDocument

The request was rejected because the policy document was malformed. The error message describes the specific error.

HTTP Status Code: 400

PackedPolicyTooLarge

The request was rejected because the total packed size of the session policies and session tags combined was too large. An AWS conversion compresses the session policy document, session policy ARNs, and session tags into a packed binary format that has a separate limit. The error message indicates by percentage how close the policies and tags are to the upper size limit. For more information, see Passing Session Tags in AWS STS in the *IAM User Guide*.

You could receive this error even though you meet other defined session policy and session tag limits. For more information, see IAM and AWS STS Entity Character Limits in the *IAM User Guide*.

HTTP Status Code: 400

RegionDisabled

AWS STS is not activated in the requested region for the account that is being asked to generate credentials. The account administrator must use the IAM console to activate AWS STS in that region. For more information, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

HTTP Status Code: 403

Examples

Example

This example illustrates one usage of GetFederationToken.

Sample Request

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetFederationToken
&Name=Bob
&PolicyArns.member.1.arn=arn:aws:iam::123456789012:policy/federateduserdemopolicy1
&PolicyArns.member.2.arn=arn:aws:iam::123456789012:policy/federateduserdemopolicy2
&Policy=%7B%22Version%22%3A%222012-10-17%22%2C%22Statement%22%3A%5B%7B%22Sid%22%3A
%22Stmt1%22%2C%22Effect%22%
 3A%22Allow%22%2C%22Action%22%3A%22s3%3A*%22%2C%22Resource%22%3A%22*%22%7D
 %5D%7D
&DurationSeconds=3600
```

```
&Tags.member.1.Key=Dept&Tags.member.1.Value=Accounting
&Tags.member.2.Key=Cost-Center&Tags.member.2.Value=12345
&AUTHPARAMS
```

Sample Response

```
<GetFederationTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetFederationTokenResult>
    <Credentials>
      <SessionToken>
       AQoDYXdzEPT//////////wEXAMPLEtc764bNrC9SAPBSM22wDOk4x4HIZ8j4FZTwdQW
       LWsKWHGBuFqwAeMicRXmxfpSPfIeoIYRqTflfKD8YUuwthAx7mSEI/qkPpKPi/kMcGd
       QrmGdeehM4IC1NtBmUpp2wUE8phUZampKsburEDy0KPkyQDYwT7WZ0wq5VSXDvp75YU
       9HFvlRd8Tx6q6fE8YQcHNVXAkiY9q6d+xo0rKwT38xVqr7ZD0u0iPPkUL64lIZbqBAz
       +scqKmlzm8FDrypNC9Yjc8fPOLn9FX9KSYvKTr4rvx3iSIlTJabIQwj2ICCR/oLxBA==
      </SessionToken>
      <SecretAccessKey>
       wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
      </SecretAccessKey>
      <Expiration>2011-07-15T23:28:33.359Z</Expiration>
      <AccessKeyId>ASIAIOSFODNN7EXAMPLE</AccessKeyId>
    </Credentials>
    <FederatedUser>
      <Arn>arn:aws:sts::123456789012:federated-user/Bob</Arn>
      <FederatedUserId>123456789012:Bob</FederatedUserId>
    </FederatedUser>
    <PackedPolicySize>6</PackedPolicySize>
  </GetFederationTokenResult>
  <ResponseMetadata>
    <RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
  </ResponseMetadata>
</GetFederationTokenResponse>
```

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

GetSessionToken

Returns a set of temporary credentials for an AWS account or IAM user. The credentials consist of an access key ID, a secret access key, and a security token. Typically, you use `GetSessionToken` if you want to use MFA to protect programmatic calls to specific AWS API operations like Amazon EC2 `StopInstances`.

MFA-enabled IAM users must call `GetSessionToken` and submit an MFA code that is associated with their MFA device. Using the temporary security credentials that the call returns, IAM users can then make programmatic calls to API operations that require MFA authentication. An incorrect MFA code causes the API to return an access denied error. For a comparison of `GetSessionToken` with the other API operations that produce temporary credentials, see Requesting Temporary Security Credentials and Comparing the AWS STS API operations in the *IAM User Guide*.

> **Note**
>
> No permissions are required for users to perform this operation. The purpose of the `sts:GetSessionToken` operation is to authenticate the user using MFA. You cannot use policies to control authentication operations. For more information, see Permissions for GetSessionToken in the *IAM User Guide*.

Session Duration

The `GetSessionToken` operation must be called by using the long-term AWS security credentials of an IAM user. Credentials that are created by IAM users are valid for the duration that you specify. This duration can range from 900 seconds (15 minutes) up to a maximum of 129,600 seconds (36 hours), with a default of 43,200 seconds (12 hours). Credentials based on account credentials can range from 900 seconds (15 minutes) up to 3,600 seconds (1 hour), with a default of 1 hour.

Permissions

The temporary security credentials created by `GetSessionToken` can be used to make API calls to any AWS service with the following exceptions:

- You cannot call any IAM API operations unless MFA authentication information is included in the request.
- You cannot call any AWS STS API *except* `AssumeRole` or `GetCallerIdentity`.

The credentials that `GetSessionToken` returns are based on permissions associated with the IAM user whose credentials were used to call the operation. The temporary credentials have the same permissions as the IAM user.

> **Note**
>
> Although it is possible to call `GetSessionToken` using the security credentials of an AWS account root user rather than an IAM user, we do not recommend it. If `GetSessionToken` is called using root user credentials, the temporary credentials have root user permissions. For more information, see Safeguard your root user credentials and don't use them for everyday tasks in the *IAM User Guide*

For more information about using `GetSessionToken` to create temporary credentials, see Temporary Credentials for Users in Untrusted Environments in the *IAM User Guide*.

Request Parameters

For information about the parameters that are common to all actions, see Common Parameters.

DurationSeconds

The duration, in seconds, that the credentials should remain valid. Acceptable durations for IAM user sessions range from 900 seconds (15 minutes) to 129,600 seconds (36 hours), with 43,200 seconds (12 hours) as the default. Sessions for AWS account owners are restricted to a maximum of 3,600 seconds (one hour). If the duration is longer than one hour, the session for AWS account owners defaults to one hour.

Type: Integer

Valid Range: Minimum value of 900. Maximum value of 129600.

Required: No

SerialNumber

The identification number of the MFA device that is associated with the IAM user who is making the `GetSessionToken` call. Specify this value if the IAM user has a policy that requires MFA authentication. The value is either the serial number for a hardware device (such as `GAHT12345678`) or an Amazon Resource Name (ARN) for a virtual device (such as

`arn:aws:iam::123456789012:mfa/user`). You can find the device for an IAM user by going to the AWS Management Console and viewing the user's security credentials.

The regex used to validate this parameter is a string of characters consisting of upper- and lower-case alphanumeric characters with no spaces. You can also include underscores or any of the following characters: =,.@:/-

Type: String

Length Constraints: Minimum length of 9. Maximum length of 256.

Pattern: `[\w+=/:,.@-]*`

Required: No

TokenCode

The value provided by the MFA device, if MFA is required. If any policy requires the IAM user to submit an MFA code, specify this value. If MFA authentication is required, the user must provide a code when requesting a set of temporary security credentials. A user who fails to provide the code receives an "access denied" response when requesting resources that require MFA authentication.

The format for this parameter, as described by its regex pattern, is a sequence of six numeric digits.

Type: String

Length Constraints: Fixed length of 6.

Pattern: `[\d]*`

Required: No

Response Elements

The following element is returned by the service.

Credentials

The temporary security credentials, which include an access key ID, a secret access key, and a security (or session) token.

> **Note**
>
> The size of the security token that AWS STS API operations return is not fixed. We strongly recommend that you make no assumptions about the maximum size.

Type: Credentials object

Errors

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

RegionDisabled

AWS STS is not activated in the requested region for the account that is being asked to generate credentials. The account administrator must use the IAM console to activate AWS STS in that region. For more information, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

HTTP Status Code: 403

Examples

Example

This example illustrates one usage of GetSessionToken.

Sample Request

```
https://sts.amazonaws.com/
?Version=2011-06-15
&Action=GetSessionToken
&DurationSeconds=3600
&Tags.member.1.Key=Project
&Tags.member.1.Value=Unicorn
&Tags.member.2.Key=Cost-Center
&Tags.member.2.Value=12345
&SerialNumber=YourMFADeviceSerialNumber
&TokenCode=123456
```

```
&AUTHPARAMS
```

Sample Response

```
<GetSessionTokenResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <GetSessionTokenResult>
    <Credentials>
      <SessionToken>
       AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT+FvwqnKwRcOIfrRh3c/L
       To6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/IvU1dYUg2RVAJBanLiHb4IgRmpRV3z
       rkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/AXlzBBko7b15fjrBs2+cTQtp
       Z3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE
      </SessionToken>
      <SecretAccessKey>
      wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY
      </SecretAccessKey>
      <Expiration>2011-07-11T19:55:29.611Z</Expiration>
      <AccessKeyId>ASIAIOSFODNN7EXAMPLE</AccessKeyId>
    </Credentials>
  </GetSessionTokenResult>
  <ResponseMetadata>
    <RequestId>58c5dbae-abef-11e0-8cfe-09039844ac7d</RequestId>
  </ResponseMetadata>
</GetSessionTokenResponse>
```

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

Data Types

The AWS Security Token Service API contains several data types that various actions use. This section describes each data type in detail.

> **Note**
>
> The order of each element in a data type structure is not guaranteed. Applications should not assume a particular order.

The following data types are supported:

- AssumedRoleUser
- Credentials
- FederatedUser
- PolicyDescriptorType
- ProvidedContext
- Tag

AssumedRoleUser

The identifiers for the temporary security credentials that the operation returns.

Contents

Arn

The ARN of the temporary security credentials that are returned from the AssumeRole action. For more information about ARNs and how to use them in policies, see IAM Identifiers in the *IAM User Guide*.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

AssumedRoleId

A unique identifier that contains the role ID and the role session name of the role that is being assumed. The role ID is generated by AWS when the role is created.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 193.

Pattern: `[\w+=,.@:-]*`

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

Credentials

AWS credentials for API authentication.

Contents

AccessKeyId

The access key ID that identifies the temporary security credentials.

Type: String

Length Constraints: Minimum length of 16. Maximum length of 128.

Pattern: `[\w]*`

Required: Yes

Expiration

The date on which the current credentials expire.

Type: Timestamp

Required: Yes

SecretAccessKey

The secret access key that can be used to sign requests.

Type: String

Required: Yes

SessionToken

The token that users must pass to the service API to use the temporary credentials.

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

FederatedUser

Identifiers for the federated user that is associated with the credentials.

Contents

Arn

The ARN that specifies the federated user that is associated with the credentials. For more information about ARNs and how to use them in policies, see IAM Identifiers in the *IAM User Guide*.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

Required: Yes

FederatedUserId

The string that identifies the federated user associated with the credentials, similar to the unique ID of an IAM user.

Type: String

Length Constraints: Minimum length of 2. Maximum length of 193.

Pattern: `[\w+=,.@\:-]*`

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

PolicyDescriptorType

A reference to the IAM managed policy that is passed as a session policy for a role session or a federated user session.

Contents

arn

The Amazon Resource Name (ARN) of the IAM managed policy to use as a session policy for the role. For more information about ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces in the *AWS General Reference*.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

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

ProvidedContext

Contains information about the provided context. This includes the signed and encrypted trusted context assertion and the context provider ARN from which the trusted context assertion was generated.

Contents

ContextAssertion

The signed and encrypted trusted context assertion generated by the context provider. The trusted context assertion is signed and encrypted by AWS STS.

Type: String

Length Constraints: Minimum length of 4. Maximum length of 2048.

Required: No

ProviderArn

The context provider ARN from which the trusted context assertion was generated.

Type: String

Length Constraints: Minimum length of 20. Maximum length of 2048.

Pattern: `[\u0009\u000A\u000D\u0020-\u007E\u0085\u00A0-\uD7FF\uE000-\uFFFD\u10000-\u10FFFF]+`

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

Tag

You can pass custom key-value pair attributes when you assume a role or federate a user. These are called session tags. You can then use the session tags to control access to resources. For more information, see Tagging AWS STS Sessions in the *IAM User Guide*.

Contents

Key

The key for a session tag.

You can pass up to 50 session tags. The plain text session tag keys can't exceed 128 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

Type: String

Length Constraints: Minimum length of 1. Maximum length of 128.

Pattern: `[\p{L}\p{Z}\p{N}_.:/=+\-@]+`

Required: Yes

Value

The value for a session tag.

You can pass up to 50 session tags. The plain text session tag values can't exceed 256 characters. For these and additional limits, see IAM and AWS STS Character Limits in the *IAM User Guide*.

Type: String

Length Constraints: Minimum length of 0. Maximum length of 256.

Pattern: `[\p{L}\p{Z}\p{N}_.:/=+\-@]*`

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