AWS Single Sign-On

OIDC API Reference

API Version 2019-06-10

AWS Single Sign-On: OIDC API Reference

Table of Contents

Welcome 1
Actions 2
CreateToken 3
Request Syntax 3
URI Request Parameters 3
Request Body 3
Response Syntax 4
Response Elements 5
Errors 5
See Also 6
RegisterClient 8
Request Syntax 8
URI Request Parameters 8
Request Body 8
Response Syntax 8
Response Elements 9
Errors 9
See Also 10
StartDeviceAuthorization 11
Request Syntax 11
URI Request Parameters 11
Request Body 11
Response Syntax 11
Response Elements 12
Errors 12
See Also 13
Data Types 14
Common Parameters 15
Common Errors 17

Welcome

AWS Single Sign-On (SSO) OpenID Connect (OIDC) is a web service that enables a client (such as AWS CLI or a native application) to register with AWS SSO. The service also enables the client to fetch the user's access token upon successful authentication and authorization with AWS SSO.

Considerations for Using This Guide

Before you begin using this guide, we recommend that you first review the following important information about how the AWS SSO OIDC service works.

- The AWS SSO OIDC service currently implements only the portions of the OAuth 2.0 Device Authorization Grant standard (https://tools.ietf.org/html/rfc8628) that are necessary to enable SSO authentication with the AWS CLI. Support for other OIDC flows frequently needed for native applications, such as Authorization Code Flow (+ PKCE), will be addressed in future releases.
- The service emits only OIDC access tokens, such that obtaining a new token (For example, token refresh) requires explicit user re-authentication.
- The access tokens provided by this service grant access to all AWS account entitlements assigned to an SSO user, not just a particular application.
- The documentation in this guide does not describe the mechanism to convert the access token into AWS Auth ("sigv4") credentials for use with IAM-protected AWS service endpoints. For more information, see GetRoleCredentials in the *AWS Single Sign-On Portal API Reference Guide*.

For general information about AWS SSO, see What is AWS Single Sign-On? in the *AWS SSO User Guide*.

This document was last published on June 6, 2022.

Actions

The following actions are supported:

- CreateToken (p. 3)
- RegisterClient (p. 8)
- StartDeviceAuthorization (p. 11)

CreateToken

Creates and returns an access token for the authorized client. The access token issued will be used to fetch short-term credentials for the assigned roles in the AWS account.

Request Syntax

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

{
   "clientId": "string",
   "clientSecret": "string",
   "code": "string",
   "deviceCode": "string",
   "grantType": "string",
   "redirectUri": "string",
   "refreshToken": "string",
   "scope": [ "string" ]
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

clientId (p. 3)

The unique identifier string for each client. This value should come from the persisted result of the RegisterClient (p. 8) API.

Type: String

Required: Yes

clientSecret (p. 3)

A secret string generated for the client. This value should come from the persisted result of the RegisterClient (p. 8) API.

Type: String

Required: Yes

code (p. 3)

The authorization code received from the authorization service. This parameter is required to perform an authorization grant request to get access to a token.

Type: String

Required: No

deviceCode (p. 3)

Used only when calling this API for the device code grant type. This short-term code is used to identify this authentication attempt. This should come from an in-memory reference to the result of the StartDeviceAuthorization (p. 11) API.

Type: String

Required: Yes

grantType (p. 3)

Supports grant types for the authorization code, refresh token, and device code request. For device code requests, specify the following value:

`urn:ietf:params:oauth:grant-type:device_code`

For information about how to obtain the device code, see the StartDeviceAuthorization (p. 11) topic.

Type: String

Required: Yes

redirectUri (p. 3)

The location of the application that will receive the authorization code. Users authorize the service to send the request to this location.

Type: String

Required: No

refreshToken (p. 3)

Currently, `refreshToken` is not yet implemented and is not supported. For more information about the features and limitations of the current AWS SSO OIDC implementation, see *Considerations for Using this Guide* in the AWS SSO OIDC API Reference.

The token used to obtain an access token in the event that the access token is invalid or expired.

Type: String

Required: No

scope (p. 3)

The list of scopes that is defined by the client. Upon authorization, this list is used to restrict permissions when granting an access token.

Type: Array of strings

Required: No

Response Syntax

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

{
   "accessToken": "string",
   "expiresIn": number,
   "idToken": "string",
   "refreshToken": "string",
   "tokenType": "string"
}
```

Response Elements

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

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

accessToken (p. 4)

An opaque token to access AWS SSO resources assigned to a user.

Type: String

expiresIn (p. 4)

Indicates the time in seconds when an access token will expire.

Type: Integer

idToken (p. 4)

Currently, `idToken` is not yet implemented and is not supported. For more information about the features and limitations of the current AWS SSO OIDC implementation, see *Considerations for Using this Guide* in the AWS SSO OIDC API Reference.

The identifier of the user that associated with the access token, if present.

Type: String

refreshToken (p. 4)

Currently, `refreshToken` is not yet implemented and is not supported. For more information about the features and limitations of the current AWS SSO OIDC implementation, see *Considerations for Using this Guide* in the AWS SSO OIDC API Reference.

A token that, if present, can be used to refresh a previously issued access token that might have expired.

Type: String

tokenType (p. 4)

Used to notify the client that the returned token is an access token. The supported type is `BearerToken`.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 17).

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

AuthorizationPendingException

Indicates that a request to authorize a client with an access user session token is pending.

HTTP Status Code: 400

ExpiredTokenException

Indicates that the token issued by the service is expired and is no longer valid.

HTTP Status Code: 400

InternalServerException

Indicates that an error from the service occurred while trying to process a request.

HTTP Status Code: 500

InvalidClientException

Indicates that the `clientId` or `clientSecret` in the request is invalid. For example, this can occur when a client sends an incorrect `clientId` or an expired `clientSecret`.

HTTP Status Code: 401

InvalidGrantException

Indicates that a request contains an invalid grant. This can occur if a client makes a CreateToken (p. 3) request with an invalid grant type.

HTTP Status Code: 400

InvalidRequestException

Indicates that something is wrong with the input to the request. For example, a required parameter might be missing or out of range.

HTTP Status Code: 400

InvalidScopeException

Indicates that the scope provided in the request is invalid.

HTTP Status Code: 400

SlowDownException

Indicates that the client is making the request too frequently and is more than the service can handle.

HTTP Status Code: 400

UnauthorizedClientException

Indicates that the client is not currently authorized to make the request. This can happen when a `clientId` is not issued for a public client.

HTTP Status Code: 400

UnsupportedGrantTypeException

Indicates that the grant type in the request is not supported by the service.

HTTP Status Code: 400

See Also

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

- AWS Command Line Interface
- AWS SDK for .NET

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

RegisterClient

Registers a client with AWS SSO. This allows clients to initiate device authorization. The output should be persisted for reuse through many authentication requests.

Request Syntax

```
POST /client/register HTTP/1.1
Content-type: application/json

{
   "clientName": "string",
   "clientType": "string",
   "scopes": [ "string" ]
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

clientName (p. 8)

The friendly name of the client.

Type: String

Required: Yes

clientType (p. 8)

The type of client. The service supports only `public` as a client type. Anything other than public will be rejected by the service.

Type: String

Required: Yes

scopes (p. 8)

The list of scopes that are defined by the client. Upon authorization, this list is used to restrict permissions when granting an access token.

Type: Array of strings

Required: No

Response Syntax

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

{
```

```
   "authorizationEndpoint": "string",
   "clientId": "string",
   "clientIdIssuedAt": number,
   "clientSecret": "string",
   "clientSecretExpiresAt": number,
   "tokenEndpoint": "string"
}
```

Response Elements

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

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

authorizationEndpoint (p. 8)

The endpoint where the client can request authorization.

Type: String

clientId (p. 8)

The unique identifier string for each client. This client uses this identifier to get authenticated by the service in subsequent calls.

Type: String

clientIdIssuedAt (p. 8)

Indicates the time at which the `clientId` and `clientSecret` were issued.

Type: Long

clientSecret (p. 8)

A secret string generated for the client. The client will use this string to get authenticated by the service in subsequent calls.

Type: String

clientSecretExpiresAt (p. 8)

Indicates the time at which the `clientId` and `clientSecret` will become invalid.

Type: Long

tokenEndpoint (p. 8)

The endpoint where the client can get an access token.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 17).

InternalServerException

Indicates that an error from the service occurred while trying to process a request.

HTTP Status Code: 500

InvalidClientMetadataException

Indicates that the client information sent in the request during registration is invalid.

HTTP Status Code: 400

InvalidRequestException

Indicates that something is wrong with the input to the request. For example, a required parameter might be missing or out of range.

HTTP Status Code: 400

InvalidScopeException

Indicates that the scope provided in the request is invalid.

HTTP Status Code: 400

See Also

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

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

StartDeviceAuthorization

Initiates device authorization by requesting a pair of verification codes from the authorization service.

Request Syntax

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

{
   "clientId": "string",
   "clientSecret": "string",
   "startUrl": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

clientId (p. 11)

The unique identifier string for the client that is registered with AWS SSO. This value should come from the persisted result of the RegisterClient (p. 8) API operation.

Type: String

Required: Yes

clientSecret (p. 11)

A secret string that is generated for the client. This value should come from the persisted result of the RegisterClient (p. 8) API operation.

Type: String

Required: Yes

startUrl (p. 11)

The URL for the AWS SSO user portal. For more information, see Using the User Portal in the *AWS Single Sign-On User Guide*.

Type: String

Required: Yes

Response Syntax

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

{
   "deviceCode": "string",
```

```
   "expiresIn": number,
   "interval": number,
   "userCode": "string",
   "verificationUri": "string",
   "verificationUriComplete": "string"
}
```

Response Elements

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

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

deviceCode (p. 11)

The short-lived code that is used by the device when polling for a session token.

Type: String

expiresIn (p. 11)

Indicates the number of seconds in which the verification code will become invalid.

Type: Integer

interval (p. 11)

Indicates the number of seconds the client must wait between attempts when polling for a session.

Type: Integer

userCode (p. 11)

A one-time user verification code. This is needed to authorize an in-use device.

Type: String

verificationUri (p. 11)

The URI of the verification page that takes the `userCode` to authorize the device.

Type: String

verificationUriComplete (p. 11)

An alternate URL that the client can use to automatically launch a browser. This process skips the manual step in which the user visits the verification page and enters their code.

Type: String

Errors

For information about the errors that are common to all actions, see Common Errors (p. 17).

InternalServerException

Indicates that an error from the service occurred while trying to process a request.

HTTP Status Code: 500

InvalidClientException

Indicates that the `clientId` or `clientSecret` in the request is invalid. For example, this can occur when a client sends an incorrect `clientId` or an expired `clientSecret`.

HTTP Status Code: 401

InvalidRequestException

Indicates that something is wrong with the input to the request. For example, a required parameter might be missing or out of range.

HTTP Status Code: 400

SlowDownException

Indicates that the client is making the request too frequently and is more than the service can handle.

HTTP Status Code: 400

UnauthorizedClientException

Indicates that the client is not currently authorized to make the request. This can happen when a `clientId` is not issued for a public client.

HTTP Status Code: 400

See Also

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

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

Data Types

The AWS SSO OIDC API has no separate data types.

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 Signature Version 4 Signing Process in the *Amazon Web Services General Reference*.

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 Task 2: Create a String to Sign for Signature Version 4 in the *Amazon Web Services General Reference*.

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 Handling Dates in Signature Version 4 in the *Amazon Web Services General Reference*.

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 Security Token Service, go to AWS Services That Work with IAM in the *IAM User Guide*.

Condition: If you're using temporary security credentials from the AWS Security Token Service, 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 Task 1: Create a Canonical Request For Signature Version 4 in the *Amazon Web Services General Reference*.

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

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

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