The token issuer endpoint

PDF (/pdfs/cognito/latest/developerguide/cognito-dg.pdf#token-endpoint)

Focus mode

On this page

Related resources

Amazon Cognito user pools API Reference (https://docs.aws.amazon.com/cognito-user-identity-pools/latest/APIReference/index.html)
AWS CLI commands for Amazon Cognito user pools (https://docs.aws.amazon.com/cli/latest/referen idp/)
SDKs & Tools (https://aws.amazon.com/tools/)

Recommended tasks

Learn about

Understand OAuth 2.0 grants for Amazon Cognito user pools
(https://docs.aws.amazon.com/cognito/latest/developerguide/federation-endpoints-oauth-grants.htm

How to

Configure Amazon Cognito to authorize REST APIs
(https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-integrate-with-cognito.

Configure a COGNITO_USER_POOLS authorizer for a REST API (https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-enable-cognito-user-po

▶ **Recently added to this guide**

Did this page help you?

Yes No

Provide feedback (https://docs.aws.amazon.com/forms/aws-doc-feedback?hidden_service_name=Cognito&topic_url=https://docs.aws.amazon.com/en_us/cognito/latest/developerg endpoint.html)

The OAuth 2.0 token endpoint (https://www.rfc-editor.org/rfc/rfc6749#section-3.2) a `/oauth2/token` issues JSON web tokens (JWTs) to applications that want to com authorization-code and client-credentials grant flows. These tokens are the end res authentication with a user pool. They contain information about the user (ID token) user's level of access (access token), and the user's entitlement to persist their signe session (refresh token). OpenID Connect (OIDC) relying-party libraries handle reque response payloads from this endpoint. Tokens provide verifiable proof of authentica profile information, and a mechanism for access to back-end systems.

Your user pool OAuth 2.0 authorization server issues JSON web tokens (JWTs) from token endpoint to the following types of sessions:

1. Users who have completed a request for an authorization code grant. Successfu redemption of a code returns ID, access, and refresh tokens.
2. Machine-to-machine (M2M) sessions that have completed a client-credentials g Successful authorization with the client secret returns an access token.
3. Users who have previously signed in and received refresh tokens. Refresh token authentication returns new ID and access tokens.

> **Note**
>
> Users who sign in with an authorization code grant in managed login or through federation can always refresh their tokens from the token endp Users who sign in with the API operations `InitiateAuth` and `AdminInitiateAuth` can refresh their tokens with the token endpoint remembered devices (./amazon-cognito-user-pools-device-tracking.html) is *n* active in your user pool. If remembered devices is active, refresh tokens the relevant API or SDK token-refresh operation (./amazon-cognito-user-po using-the-refresh-token.html#using-the-refresh-token-api) for your app client.

The token endpoint becomes publicly available when you add a domain to your use accepts HTTP POST requests. For application security, use PKCE with your authoriza sign-in events. PKCE verifies that the user passing an authorization code is that sam who authenticated. For more information about PKCE, see IETF RFC 7636 (https://datatracker.ietf.org/doc/html/rfc7636) .

You can learn more about the user pool app clients and their grant types, client secr allowed scopes, and client IDs at Application-specific settings with app clients (./user settings-client-apps.html) . You can learn more about M2M authorization, client creder grants, and authorization with access token scopes at Scopes, M2M, and APIs with re servers (./cognito-user-pools-define-resource-servers.html) .

To retrieve information about a user from their access token, pass it to your userInfo endpoint (./userinfo-endpoint.html) or to a GetUser (https://docs.aws.amazon.com/cognit identity-pools/latest/APIReference/API_GetUser.html) API request. The access token mus the appropriate scopes for these requests,

Format a POST request to the token endpoint

The `/oauth2/token` endpoint only supports `HTTPS POST`. This endpoint is not u interactive. Handle token requests with an OpenID Connect (OIDC) library (https://openid.net/developers/certified-openid-connect-implementations/) in your applica

The token endpoint supports `client_secret_basic` and `client_secret_pos` authentication. For more information about the OIDC specification, see Client Authe (https://openid.net/specs/openid-connect-core-1_0.html#ClientAuthentication) . For mo information about the token endpoint from the OpenID Connect specification, see T Endpoint (http://openid.net/specs/openid-connect-core-1_0.html#TokenEndpoint) .

Request parameters in header

You can pass the following parameters in the header of your request to the token e

`Authorization`

If the client was issued a secret, the client can pass its `client_id` and `client_secret` in the authorization header as `client_secret_basic` HTTP authorization. You can also include the `client_id` and `client_secret` in the request body as `client_secret_post` authorization.

The authorization header string is Basic (https://en.wikipedia.org/wiki/Basic_access_authentication#Client_side) `Base64Encode(client_id:client_secret)`. The following example is an authorization header for app client `djc98u3jiedmi283eu928` with client secre `abcdef01234567890`, using the Base64-encoded version of the string `djc98u3jiedmi283eu928:abcdef01234567890`:

```
Authorization: Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw
```

`Content-Type`

Set the value of this parameter to `'application/x-www-form-urlencoded'`

Request parameters in body

The following are parameters that you can request in `x-www-form-urlencoded` f the request body to the token endpoint.

`grant_type`

Required.

The type of OIDC grant that you want to request.

Must be `authorization_code` or `refresh_token` or `client_credential` can request an access token for a custom scope from the token endpoint under t following conditions:

- You enabled the requested scope in your app client configuration.
- You configured your app client with a client secret.
- You enable client credentials grant in your app client.

> **Note**
> The token endpoint returns a refresh token only when the `grant_type` i `authorization_code`.

`client_id`

Optional. Not required when you provide the app client ID in the `Authorization` header.

The ID of an app client in your user pool. Specify the same app client that authen your user.

You must provide this parameter if the client is public and does not have a secret with `client_secret` in `client_secret_post` authorization.

`client_secret`

Optional. Not required when you provide the client secret in the `Authorization` and when the app client doesn't have a secret.

The app client secret, if the app client has one, for `client_secret_post` authorization.

`scope`

Optional.

Can be a combination of any scopes that are associated with your app client. Ama Cognito ignores scopes in the request that aren't allowed for the requested app c you don't provide this request parameter, the authorization server returns an acc token `scope` claim with all authorization scopes that you enabled in your app cl configuration. You can request any of the scopes allowed for the requested app c standard scopes, custom scopes from resource servers, and the `aws.cognito.signin.user.admin` user self-service scope.

`redirect_uri`

Optional. Not required for client-credentials grants.

Must be the same `redirect_uri` that was used to get `authorization_code` `/oauth2/authorize`.

You must provide this parameter if `grant_type` is `authorization_code`.

`refresh_token`

Optional. Used only when the user already has a refresh token and wishes to get ne and access tokens.

To generate new access and ID tokens for a user's session, set the value of `refresh_token` to a valid refresh token that the requested app client issued.

Returns a new refresh token with new ID and access token when refresh token ro (./amazon-cognito-user-pools-using-the-refresh-token.html#using-the-refresh-token-rota active, otherwise returns only ID and access tokens. If the original access token w bound to an API resource (./cognito-user-pools-define-resource-servers.html#cognito-u pools-resource-binding) , the new access token maintains the requested API url in t claim.

code

Optional. Only required in authorization-code grants.

The authorization code from an authorization code grant. You must provide this parameter if your authorization request included a `grant_type` of `authorization_code`.

aws_client_metadata

Optional.

Information that you want to pass to the Pre token generation Lambda trigger (./ pool-lambda-pre-token-generation.html) in machine-to-machine (M2M) (./cognito-use define-resource-servers.html) authorization flows. Your application can collect conte information about the session and pass it in this parameter. When you pass `aws_client_metadata` in URL-encoded JSON format, Amazon Cognito includ the input event to your trigger Lambda function. Your pre token trigger event ve global Lambda trigger version must be configured for version three or later. Alth Amazon Cognito accepts requests to this endpoint in authorization code and clie credentials M2M flows, your user pool only passes `aws_client_metadata` to t token generation trigger from client credentials requests.

code_verifier

Optional. Required only if you provided `code_challenge_method` and `code_challenge` parameters in your initial authorization request.

The generated code verifier that your application calculated the `code_challen` from in an authorization code grant request with PKCE (./using-pkce-in-authorizatio code.html) .

Exchanging an authorization code for tokens

The following request successfully generates ID, access, and refresh tokens after authentication with an authorization-code grant. The request passes the client secr `client_secret_basic` format in the `Authorization` header.

```
POST https://mydomain.auth.us-east-
1.amazoncognito.com/oauth2/token&
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=authorization_code&
client_id=1example23456789&
```

```
code=AUTHORIZATION_CODE&
redirect_uri=com.myclientapp://myclient/redirect
```

The response issues new ID, access, and refresh tokens to the user, with additional m

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

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Client credentials with basic authorization

The following request from an M2M application requests a client credentials grant. I
client credentials requires a client secret, the request is authorized with an Author
header derived from the app client ID and secret. The request results in an access to
the two requested scopes. The request also includes client metadata that provides I
information and a token issued to the user who this grant is on behalf of. Amazon C
passes the client metadata to the pre token generation Lambda trigger.

```
POST https://mydomain.auth.us-east-
1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=client_credentials&
client_id=1example23456789&
scope=resourceServerIdentifier1%2Fscope1%20resourceServerIden
fier2%2Fscope2&
&aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789gh
XAMPLE%22,%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

Amazon Cognito passes the following input event to the pre token generation Lamb
trigger.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/scope1',
           'resourceServerIdentifier2/scope2'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

The response returns an access token. Client credentials grants are for machine-to-r
(M2M) authorization and only return access tokens.

```
HTTP/1.1 200 OK
Content-Type: application/json
{
    "access_token": "eyJra1example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Client credentials with POST body authorization

The following client-credentials grant request includes the `client_secret` param
the request body and doesn't include an `Authorization` header. This request use
`client_secret_post` authorization syntax. The request results in an access toke
the requested scope. The request also includes client metadata that provides IP-add
information and a token issued to the user who this grant is on behalf of. Amazon C
passes the client metadata to the pre token generation Lambda trigger.

```
POST /oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
X-Amz-Target: AWSCognitoIdentityProviderService.Client
credentials request
User-Agent: USER_AGENT
Accept: /
Accept-Encoding: gzip, deflate, br
Content-Length: 177
Referer: http://auth.example.com/oauth2/token
Host: auth.example.com
Connection: keep-alive
```

```
grant_type=client_credentials&
client_id=1example23456789&
scope=my_resource_server_identifier%2Fmy_custom_scope&
client_secret=9example87654321&
aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789ghi
AMPLE%22,%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

Amazon Cognito passes the following input event to the pre token generation Lamb
trigger.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/my_custom_scope'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

The response returns an access token. Client credentials grants are for machine-to-r
(M2M) authorization and only return access tokens.

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Date: Tue, 05 Dec 2023 16:11:11 GMT
x-amz-cognito-request-id: 829f4fe2-a1ee-476e-b834-5cd85c03373

{
    "access_token": "eyJra12345EXAMPLE",
    "expires_in": 3600,
    "token_type": "Bearer"
}
```

Authorization code grant with PKCE

The following example request completes an authorization request that included `code_challenge_method` and `code_challenge` parameters in an authorizatio grant request with PKCE (./using-pkce-in-authorization-code.html) .

```
POST https://mydomain.auth.us-east-
1.amazoncognito.com/oauth2/token
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=authorization_code&
client_id=1example23456789&
code=AUTHORIZATION_CODE&
code_verifier=CODE_VERIFIER&
redirect_uri=com.myclientapp://myclient/redirect
```

The response returns ID, access, and refresh tokens from the successful PKCE verific the application.

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

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Token refresh without refresh token rotation

The following example requests provides a refresh token to an app client where refr token rotation (./amazon-cognito-user-pools-using-the-refresh-token.html#using-the-refre rotation) is inactive. Because the app client has a client secret, the request provides a `Authorization` header.

```
POST https://mydomain.auth.us-east-
1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=refresh_token&
client_id=1example23456789&
refresh_token=eyJj3example
```

The response returns new ID and access tokens.

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

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Token refresh with refresh token rotation

The following example requests provides a refresh token to an app client where refr token rotation (./amazon-cognito-user-pools-using-the-refresh-token.html#using-the-refre rotation) is active. Because the app client has a client secret, the request provides an `Authorization` header.

```
POST https://mydomain.auth.us-east-
1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic
ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=refresh_token&
client_id=1example23456789&
refresh_token=eyJj3example
```

The response returns new ID, access, and refresh tokens.

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

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj4example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Examples of negative responses

Malformed requests generate errors from the token endpoint. The following is a ge map of the response body when token requests generate an error.

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8
```

```
{
"error":"invalid_request|invalid_client|invalid_grant|unautho
zed_client|unsupported_grant_type"
}
```

`invalid_request`

The request is missing a required parameter, includes an unsupported parameter (other than `unsupported_grant_type`), or is otherwise malformed. For examp `grant_type` is `refresh_token` but `refresh_token` is not included.

`invalid_client`

Client authentication failed. For example, when the client includes `client_id` a `client_secret` in the authorization header, but there's no such client with tha `client_id` and `client_secret`.

`invalid_grant`

Refresh token has been revoked.

Authorization code has been consumed already or does not exist.

App client doesn't have read access to all attributes (https://docs.aws.amazon.com/cognito/latest/developerguide/user-pool-settings-attribut in the requested scope. For example, your app requests the `email` scope and you client can read the `email` attribute, but not `email_verified`.

`unauthorized_client`

Client is not allowed for code grant flow or for refreshing tokens.

`unsupported_grant_type`

Returned if `grant_type` is anything other than `authorization_code` or `refresh_token` or `client_credentials`.

View related pages *Abstracts generated by AI*

Glue › dg
Configuring Intercom connections...
Configuring Intercom connections involves creating connected apps, providing client credentials, storing secrets in AWS Secrets Manager, and granting IAM role permissions to access secrets for AWS Glue connections.
December 18, 2024

Cognito › developerguide
The redirect and authorization endpoi...
Amazon Cognito authorization server handles user authentication, authorization code grants, implicit grants, re-authentication, silent authentication, user profile scopes, and authorization server errors.
October 26, 2025

Verifiedpermissions ›.
Client and audier
validation for Am
Verified Permission
validates Amazon C
ID, access tokens, C
client IDs, audience
Parses JWT claims
client-side authoriz
October 26, 2025

Discover highly rated pages *Abstracts generated by AI*

Cognito › developerguide
What is Amazon Cognito?...
Amazon Cognito authenticates users, authorizes AWS resource access, issues temporary AWS credentials, integrates with identity providers, manages user pools and identity pools, configures role-based access control.
October 26, 2025

Cognito › developerguide
Understanding user pool JSON web token...
Authenticating users with tokens, storing tokens securely, customizing access and ID tokens, understanding ID token claims, understanding access token claims, refreshing tokens for new access, revoking tokens to end sessions.
October 26, 2025

Cognito › developergu
Common Amazo
Cognito scenario
Amazon Cognito er
user authenticatior
access to back-end
resources, AWS ser
via API Gateway, La
identity pools, thirc
IdPs, and AppSync
resources.
October 26, 2025