OpenID Connect Authentication

- Configuration Example
- Configuration Options
  - redirectUrl
    - Add specific rule on the IngressRoute
    - Inherit the Protocol and Domain from the Request and Uses the Redirecturl’s Path
    - Inherit the Protocol from the Request and Uses the Redirecturl’s Domain and Path
    - Replace the Request URL with the Redirect URL since It Is an Absolute URL
  - clientId, clientSecret
    - Storing secret values in Kubernetes secrets
  - claims
    - Syntax
    - Nested Claims
  - clientConfig
    - clientConfig.tls
      - Storing secret values in Kubernetes secrets
  - sameSite - Accepted values
  - session.store
  - csrf
    - CSRF Internal Behavior

!!! info "Traefik Hub Feature" This middleware is available exclusively in Traefik Hub. Learn more about Traefik Hub's advanced features.

The OIDC Authentication middleware secures your applications by delegating the authentication to an external provider

---

Configuration Example

```yaml
apiVersion: traefik.io/v1alpha1
kind: Middleware
metadata:
  name: test-oidc
  namespace: whoami
spec:
  plugin:
    oidc:
      issuer: "https://tenant.auth0.com/realms/myrealm"
      redirectUrl: "/callback"
      clientId: "urn:k8s:secret:my-secret:clientId"
      clientSecret: "urn:k8s:secret:my-secret:clientSecret"
      session:
        name: customsessioncookiename
        sliding: false
        refresh: false
        expiry: 10
        sameSite: none
        httpOnly: false
        secure: true
      stateCookie:
        name: customstatecookiename
        maxAge: 10
        sameSite: none
        httpOnly: true
        secure: true
      forwardHeaders:
        Group: grp
        Expires-At: exp
      claims: Equals(`grp`, `admin`)
      csrf: {}
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: my-secret
```

```
stringData:
  clientID: my-oidc-client-name
  clientSecret: mysecret
```

Configuration Options

Field	Description	Default	Required
`issuer`	Defines the URL to the OpenID Connect provider (for example, `https://accounts.google.com`). It should point to the server which provides the OpenID Connect configuration.	""	Yes
`trustedIssuer`	Defines a trusted issuer URL to validate against in addition to the one discovered from the OpenID Connect provider.	""	No
`disableIssuerCheck`	Disables the issuer validation check during token verification.	false	No
`redirectUrl`	Defines the URL used by the OpenID Connect provider to redirect back to the middleware once the authorization is complete. (More information here)	""	Yes
`clientId`	Defines the unique client identifier for an account on the OpenID Connect provider, must be set when the `clientSecret` option is set. (More information here)	""	Yes
`clientSecret`	Defines the unique client secret for an account on the OpenID Connect provider, must be set when the `clientId` option is set. (More information here)	""	Yes

Field	Description	Default	Required
`claims`	Defines the claims to validate in order to authorize the request. The `claims` option can only be used with JWT-formatted token. (More information here)	""	No
`usernameClaim`	Defines the claim that will be evaluated to populate the `clientusername` in the access logs. The `usernameClaim` option can only be used with JWT-formatted token.	""	No
`forwardHeaders`	Defines the HTTP headers to add to requests and populates them with values extracted from the access token claims returned by the authorization server. Claims to be forwarded that are not found in the JWT result in empty headers. The `forwardHeaders` option can only be used with JWT-formatted token.	[]	No
`clientConfig.tls.ca`	PEM-encoded certificate bundle or a URN referencing a secret containing the certificate bundle used to establish a TLS connection with the authorization server (More information here)	""	No
`clientConfig.tls.cert`	PEM-encoded certificate or a URN referencing a secret containing the certificate used to establish a TLS connection with the Vault server (More information here)	""	No
`clientConfig.tls.key`	PEM-encoded key or a URN referencing a secret containing the key used to establish a TLS connection with the Vault server. (More information here)	""	No

Field	Description	Default	Required
`clientConfig.tls.insecureSkipVerify`	Disables TLS certificate verification when communicating with the authorization server. Useful for testing purposes but strongly discouraged for production. (More information here)	""	No
`clientConfig.timeoutSeconds`	Defines the time before giving up requests to the authorization server.	5	No
`clientConfig.maxRetries`	Defines the number of retries for requests to authorization server that fail.	3	No
`pkce`	Defines the Proof Key for Code Exchange as described in RFC 7636.	false	No
`discoveryParams`	A map of arbitrary query parameters to be added to the openid-configuration well-known URI during the discovery mechanism.	""	No
`scopes`	The scopes to request. Must include `openid`.	openid	No
`authParams`	A map of the arbitrary query parameters to be passed to the Authentication Provider. When a `prompt` key is set to an empty string in the AuthParams,the prompt parameter is not added to the OAuth2 authorization URL Which means the user won't be prompted for consent.	""	No
`disableLogin`	Disables redirections to the authentication provider This can be useful for protecting APIs where redirecting to a login page is undesirable.	false	No

Field	Description	Default	Required
`loginUrl`	Defines the URL used to start authorization when needed. All other requests that are not already authorized will return a 401 Unauthorized. When left empty, all requests can start authorization. It can be a path (`/login` for example), a host and a path (`example.com/login`) or a complete URL (`https://example.com/login`). Only `http` and `https` schemes are supported.	""	No
`logoutUrl`	Defines the URL on which the session should be deleted in order to log users out. It can be a path (`/logout` for example), a host and a path (`example.com/logout`) or a complete URL (`https://example.com/logout`). Only `http` and `https` schemes are supported.	""	No
`postLoginRedirectUrl`	If set and used in conjunction with `loginUrl`, the middleware will redirect to this URL after successful login. It can be a path (`/after/login` for example), a host and a path (`example.com/after/login`) or a complete URL (`https://example.com/after/login`). Only `http` and `https` schemes are supported.	""	No

Field	Description	Default	Required
`postLogoutRedirectUrl`	If set and used in conjunction with `logoutUrl`, the middleware will redirect to this URL after logout. It can be a path (`/after/logout` for example), a host and a path (`example.com/after/logout`) or a complete URL (`https://example.com/after/logout`). Only `http` and `https` schemes are supported.	""	No
`backchannelLogoutUrl`	Defines the URL called by the OIDC provider when a user logs out (see https://openid.net/specs/openid-connect-rpinitiated-1_0.html#OpenID.BackChannel). It can be a path (`/backchannel-logout` for example), a host and a path (`example.com/backchannel-logout`) or a complete URL (`https://example.com/backchannel-logout`). Only `http` and `https` schemes are supported. This feature is currently in an experimental state and has been tested exclusively with the Keycloak OIDC provider.	""	No
`backchannelLogoutSessionsRequired`	This specifies whether the OIDC provider includes the sid (session ID) Claim in the Logout Token to identify the user session (see https://openid.net/specs/openid-connect-backchannel-1_0.html#BCRegistration). If omitted, the default value is false. This feature is currently in an experimental state and has been tested exclusively with the Keycloak OIDC provider.	false	No

Field	Description	Default	Required
`stateCookie.name`	Defines the name of the state cookie.	"`MIDDLEWARE_NAME`-state"	No
`stateCookie.path`	Defines the URL path that must exist in the requested URL in order to send the Cookie header. The `%x2F` ('/') character is considered a directory separator, and subdirectories will match as well. For example, if `stateCookie.path` is set to `/docs`, these paths will match: `/docs`, `/docs/web/`, `/docs/web/http`.	"/"	No
`stateCookie.domain`	Defines the hosts that are allowed to receive the cookie. If specified, then subdomains are always included. For example, if it is set to `example.com`, then cookies are included on subdomains like `api.example.com`.	""	No
`stateCookie.maxAge`	Defines the number of seconds after which the state cookie should expire. A zero or negative number will expire the cookie immediately.	600	No
`stateCookie.sameSite`	Informsbrowsers how they should handle the state cookie on cross-site requests. Setting it to `lax` or `strict` can provide some protection against cross-site request forgery attacks (CSRF). More information here.	lax	No

Field	Description	Default	Required
`stateCookie.httpOnly`	Forbids JavaScript from accessing the cookie. For example, through the `Document.cookie` property, the `XMLHttpRequest` API, or the `Request` API. This mitigates attacks against cross-site scripting (XSS).	true	No
`stateCookie.secure`	Defines whether the state cookie is only sent to the server when a request is made with the `https` scheme.	false	No
`session.name`	The name of the session cookie.	"`MIDDLEWARE_NAME`-session"	No
`session.path`	Defines the URL path that must exist in the requested URL in order to send the Cookie header. The `%x2F` ('/'') character is considered a directory separator, and subdirectories will match as well. For example, if `stateCookie.path` is set to `/docs`, these paths will match: `/docs`, `/docs/web/`, `/docs/web/http`.	"/"	No
`session.domain`	Specifies the hosts that are allowed to receive the cookie. If specified, then subdomains are always included. If specified, then subdomains are always included. For example, if it is set to `example.com`, then cookies are included on subdomains like `api.example.com`.	""	No
`session.expiry`	Number of seconds after which the session should expire. A zero or negative number is **prohibited**.	86400 (24h)	No

Field	Description	Default	Required
`session.sliding`	Forces the middleware to renew the session cookie each time an authenticated request is received.	true	No
`session.refresh`	Enables the access token refresh when it expires.	true	No
`session.sameSite`	Inform browsers how they should handle the session cookie on cross-site requests. Setting it to `lax` or `strict` can provide some protection against cross-site request forgery attacks (CSRF). More information here.	lax	No
`session.httpOnly`	Forbids JavaScript from accessing the cookie. For example, through the `Document.cookie` property, the `XMLHttpRequest` API, or the `Request` API. This mitigates attacks against cross-site scripting (XSS).	true	No
`session.secure`	Defines whether the session cookie is only sent to the server when a request is made with the `https` scheme.	false	No
`session.store.redis.endpoints`	Endpoints of the Redis instances to connect to (example: `redis.traefik-hub.svc.cluster.local:6379`)	""	Yes
`session.store.redis.username`	The username Traefik Hub will use to connect to Redis	""	No
`session.store.redis.password`	The password Traefik Hub will use to connect to Redis	""	No
`session.store.redis.database`	The database Traefik Hub will use to sore information (default: `0`)	""	No
`session.store.redis.cluster`	Enable Redis Cluster	""	No
`session.store.redis.tls.caBundle`	Custom CA bundle	""	No
`session.store.redis.tls.cert`	TLS certificate	""	No

Field	Description	Default	Required
`session.store.redis.tls.key`	TLS key	""	No
`session.store.redis.tls.insecureSkipVerify`	Allow skipping the TLS verification	""	No
`session.store.redis.sentinel.masterSet`	Name of the set of main nodes to use for main selection. Required when using Sentinel.	""	No
`session.store.redis.sentinel.username`	Username to use for sentinel authentication (can be different from `username`)	""	No
`session.store.redis.sentinel.password`	Password to use for sentinel authentication (can be different from `password`)	""	No
`session.store.keyPrefix`	Defines the prefix of the key for the entries that store the sessions.	""	No
`csrf`	When enabled, a CSRF cookie, named `hub-csrf-token`, is bound to the OIDC session to protect service from CSRF attacks. It is based on the Signed Double Submit Cookie implementation as defined by the OWASP Foundation. Moreinformation here.	""	No
`csrf.secure`	Defines whether the CSRF cookie is only sent to the server when a request is made with the `https` scheme.	false	No
`csrf.headerName`	Defines the name of the header used to send the CSRF token value received previously in the CSRF cookie.	Hub-Csrf-Token	No

redirectUrl

Add specific rule on the IngressRoute

The URL informs the OpenID Connect provider how to return to the middleware. If the router rule is accepting all paths on a domain, no extra work is needed. If the router rule is specific about the paths allowed, the path set in this option should be included.

```
apiVersion: traefik.io/v1alpha1
kind: IngressRoute
metadata:
  name: whoami
spec:
  entryPoints:
    - web
    - websecure
  routes:
      # Rules to match the loginUrl and redirectUrl can be added into
      # your current router.
    - match: Path(`/myapi`) || Path(`/login`) || Path(`/callback`)
      kind: Rule
      middlewares:
        - name: test-oidc
```

This URL will not be passed to the upstream application, but rather handled by the middleware itself. The chosen URL should therefore not conflict with any URLs needed by the upstream application.

This URL sometimes needs to be set in the OpenID Connect Provider's configuration as well (like for Google Accounts for example).

It can be the absolute URL, relative to the protocol (inherits the request protocol), or relative to the domain (inherits the request domain and protocol). See the following examples.

Inherit the Protocol and Domain from the Request and Uses the Redirecturl’s Path

Request URL	RedirectURL	Result
`http://expl.co`	`/cback`	`http://expl.co/cback`

Inherit the Protocol from the Request and Uses the Redirecturl’s Domain and Path

Request URL	RedirectURL	Result
`https://scur.co`	`expl.co/cback`	`https://expl.co/cback`

Replace the Request URL with the Redirect URL since It Is an Absolute URL

Request URL	RedirectURL	Result
`https://scur.co`	`http://expl.co/cback`	`http://expl.co/cback`

!!! note "Supported Schemes"

```
Only `http` and `https` schemes are supported.
```

```
apiVersion: traefik.io/v1alpha1
kind: Middleware
metadata:
  name: test-oidc
spec:
  plugin:
    oidc:
      issuer: "https://tenant.auth0.com/realms/myrealm"
      redirectUrl: "/callback"
      clientId: my-oidc-client-name
      clientSecret: mysecret
```

clientId, clientSecret

Storing secret values in Kubernetes secrets

When configuring the `clientId` and the `clientSecret`, it is possible to reference Kubernetes secrets defined in the same namespace as the Middleware. The reference to a Kubernetes secret takes the form of a URN:

```
urn:k8s:secret:[name]:[valueKey]
```

claims

Syntax

The following functions are supported in `claims`:

Function	Description	Example
Equals	Validates the equality of the value in `key` with `value`.	Equals(`grp`, `admin`)

Function	Description	Example
Prefix	Validates the value in `key` has the prefix of `value`.	Prefix(`referrer`, `http://example.com`)
Contains (string)	Validates the value in `key` contains `value`.	Contains(`referrer`, `/foo/`)
Contains (array)	Validates the `key` array contains the `value`.	Contains(`areas`, `home`)
SplitContains	Validates the value in `key` contains the `value` once split by the separator.	SplitContains(`scope`, ` `, `writer`)
OneOf	Validates the `key` array contains one of the `values`.	OneOf(`areas`, `office`, `lab`)

All functions can be joined by boolean operands. The supported operands are:

Operand	Description	Example
&&	Compares two functions and returns true only if both evaluate to true.	Equals(`grp`, `admin`) && Equals(`active`, `true`)
||	Compares two functions and returns true if either evaluate to true.	Equals(`grp`, `admin`) || Equals(`active`, `true`)
!	Returns false if the function is true, otherwise returns true.	!Equals(`grp`, `testers`)

All examples will return true for the following data structure:

```
{
  "active": true,
  "grp": "admin",
  "scope": "reader writer deploy",
  "referrer": "http://example.com/foo/bar",
  "areas": [
    "office",
    "home"
  ]
}
```

Nested Claims

Nested claims are supported by using a `.` between keys. For example:

```
user.name
```

```
{
  "active": true,
  "grp": "admin",
  "scope": "reader writer deploy",
  "referrer": "http://example.com/foo/bar",
  "areas": [
    "office",
    "home"
  ],
  "user" {
    "name": "John Snow",
    "status": "undead"
  }
}
```

```
John Snow
```

!!! note "Handling keys that contain a '.'"

If the `key` contains a dot, the dot can be escaped using `\.`

!!! note "Handling a key that contains a '"

If the `key` contains a `\`, it needs to be doubled `\\`.

!!! note "Access Token and ID Token claims"

```
The first argument of the function, which represents the key to look for in the token claims,
can be prefixed to specify which of the two kinds of token is inspected.
Possible prefix values are `id_token.` and `access_token.`. If no prefix is specified, it
defaults to the ID token.

| Example                                  |
Description                                                                  |
| ---------------------------------------- |
------------------------------------------------------------------------------ |
| <a id="opt-Equalsid-token-grp-admin" href="#opt-Equalsid-token-grp-admin" title="#opt-
Equalsid-token-grp-admin">Equals(\`id_token.grp\`, \`admin\`)</a> | Checks if the value of
claim `grp` in the ID token is `admin`.            |
| <a id="opt-Prefixaccess-token-referrer-httpexample-com" href="#opt-Prefixaccess-token-
referrer-httpexample-com" title="#opt-Prefixaccess-token-referrer-httpexample-
```

```
com">Prefix(\`access_token.referrer\`, \`http://example.com\`)</a> | Checks if the value of
claim `referrer` in the access token is prefixed by `http://example.com\`.|
| <a id="opt-OneOfareas-office-lab" href="#opt-OneOfareas-office-lab" title="#opt-OneOfareas-
office-lab">OneOf(\`areas\`, \`office\`, \`lab\`)</a> | Checks if the value of claim `areas`
in the ID token is `office` or `labs`.                                    |
```

clientConfig

Defines the configuration used to connect the API Gateway to a Third Party Software such as an Identity Provider.

`clientConfig.tls`

Storing secret values in Kubernetes secrets

When configuring the `tls.ca`, `tls.cert`, `tls.key`, it is possible to reference Kubernetes secrets defined in the same namespace as the Middleware.
The reference to a Kubernetes secret takes the form of a URN:

```
urn:k8s:secret:[name]:[valueKey]
```

```
apiVersion: traefik.io/v1alpha1
kind: Middleware
metadata:
  name: test-oidc
spec:
  plugin:
    oidc:
      clientConfig:
        tls:
          ca: "urn:k8s:secret:tls:ca"
          cert: "urn:k8s:secret:tls:cert"
          key: "urn:k8s:secret:tls:key"
          insecureSkipVerify: true
```

```
apiVersion: v1
kind: Secret
metadata:
  name: tls
stringData:
```

```
ca: |-
  -----BEGIN CERTIFICATE-----
  MIIB9TCCAWACAQAwgbgxGTAXBgNVBAoMEFF1b1ZhZGlzIExpbWl0ZWQxHDAaBgNV
  BAsME0RvY3VtZW50IERlcGFydG1lbnQxOTA3BgNVBAMMMFdoeSBhcmUgeW91IGRl
  Y29kaW5nIG1lPyAgVGhpcyBpcyBvbmx5IGEgdGVzdCEhITERMA8GA1UEBwwISGFt
  aWx0b24xETAPBgNVBAgMCFBlbWJyb2tlMQswCQYDVQQGEwJCTTEPMA0GCSqGSIb3
  DQEJARYAMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCJ9WRanG/fUvcfKiGl
  EL4aRLjGt537mZ28UU9/3eiJeJznNSOuNLnF+hmabAu7H0LT4K7EdqfF+XUZW/2j
  RKRYcvOUDGF9A7OjW7UfKk1In3+6QDCi7X34RE161jqoaJjrm/T18TOKcgkkhRzE
  apQnIDm0Ea/HVzX/PiSOGuertwIDAQABMAsGCSqGSIb3DQEBBQOBgQBzMJdAV4QP
  Awel8LzGx5uMOshezF/KfP67wJ93UW+N7zXY6AwPgoLj4Kjw+WtU684JL8Dtr9FX
  ozakE+8p06BpxegR4BR3FMHf6p+0jQxUEAkAyb/mVgm66TyghDGC6/YkiKoZptXQ
  98TwDIK/39WEB/V607As+KoYazQG8drorw==
  -----END CERTIFICATE-----
cert: |-
  -----BEGIN CERTIFICATE-----
  MIIB9TCCAWACAQAwgbgxGTAXBgNVBAoMEFF1b1ZhZGlzIExpbWl0ZWQxHDAaBgNV
  BAsME0RvY3VtZW50IERlcGFydG1lbnQxOTA3BgNVBAMMMFdoeSBhcmUgeW91IGRl
  Y29kaW5nIG1lPyAgVGhpcyBpcyBvbmx5IGEgdGVzdCEhITERMA8GA1UEBwwISGFt
  aWx0b24xETAPBgNVBAgMCFBlbWJyb2tlMQswCQYDVQQGEwJCTTEPMA0GCSqGSIb3
  DQEJARYAMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCJ9WRanG/fUvcfKiGl
  EL4aRLjGt537mZ28UU9/3eiJeJznNSOuNLnF+hmabAu7H0LT4K7EdqfF+XUZW/2j
  RKRYcvOUDGF9A7OjW7UfKk1In3+6QDCi7X34RE161jqoaJjrm/T18TOKcgkkhRzE
  apQnIDm0Ea/HVzX/PiSOGuertwIDAQABMAsGCSqGSIb3DQEBBQOBgQBzMJdAV4QP
  Awel8LzGx5uMOshezF/KfP67wJ93UW+N7zXY6AwPgoLj4Kjw+WtU684JL8Dtr9FX
  ozakE+8p06BpxegR4BR3FMHf6p+0jQxUEAkAyb/mVgm66TyghDGC6/YkiKoZptXQ
  98TwDIK/39WEB/V607As+KoYazQG8drorw==
  -----END CERTIFICATE-----
key: |-
  -----BEGIN EC PRIVATE KEY-----
  MHcCAQEEIC8CsJ/B115S+JtR1/l3ZQwKA3XdXt9zLqusF1VXc/KloAoGCCqGSM49
  AwEHoUQDQgAEpwUmRIZHFt8CdDHYm1ikScCScd2q6QVYXxJu+G3fQZ78ScGtN7fu
  KXMnQqVjXVRAr8qUY8yipVKuMCepnPXScQ==
  -----END EC PRIVATE KEY-----
```

sameSite - Accepted values

- `none`: Thebrowser will send cookies with both cross-site requests and same-site requests.

- `strict`: Thebrowser will only send cookies for same-site requests (requests originating from the site that set the cookie). If the request originated from a different URL than the URL of the current location, none of the cookies tagged with the `strict` attribute will be included.
- `lax`: Same-site cookies are withheld on cross-site subrequests, such as calls to load images or frames, but will be sent when a user navigates to the URL from an external site; for example, by following a link.

session.store

An OpenID Connect Authentication middleware can use a persistent KV storage to store the `HTTP` sessions data instead of keeping all the state in cookies. It avoids cookies growing inconveniently large, which can lead to latency issues.

Refer to the redis options to configure the Redis connection.

Connection parameters to your Redis server are attached to your Middleware deployment.

The following Redis modes are supported:

- Single instance mode
- Redis Cluster
- Redis Sentinel

!!! info

```
If you use Redis in single instance mode or Redis Sentinel, you can configure the `database` field.
This value won't be taken into account if you use Redis Cluster (only database `0` is available).

In this case, a warning is displayed, and the value is ignored.
```

For more information about Redis, we recommend the official Redis documentation.

```
apiVersion: traefik.io/v1alpha1
kind: Middleware
metadata:
  name: test-oidc
spec:
  plugin:
```

```
    oidc:
      issuer: "https://tenant.auth0.com/realms/myrealm"
      redirectUrl: "/callback"
      clientId: my-oidc-client-name
      clientSecret: mysecret
      session:
        store:
          redis:
            endpoints:
              - redis-master.traefik-hub.svc.cluster.local:6379
            password: "urn:k8s:secret:oidc:redisPass"
```

```
apiVersion: v1
kind: Secret
metadata:
  name: oidc
stringData:
  redisPass: mysecret12345678
```

csrf

CSRF Internal Behavior

When the OIDC session is expired, the corresponding CSRF cookie is deleted. This means that a new CSRF token will be generated and sent to the client whenever the session is refreshed or recreated.

When a request is sent and uses a non-safe method (see RFC7231#section-4.2.1), the CSRF token value (extracted from the cookie) have to be sent to the server in the header configured with the headerName option.

{% include-markdown "includes/traefik-for-business-applications.md" %}

□□□ #2
□ □□□ □□□ 2026-10-31 14:03:32 UTC
□ □□□ □□□ 2026-10-31 14:03:33 UTC