security oauth2 commands

ONTAP 9.15.1 commands

NetApp
December 18, 2024

Table of Contents

security oauth2 commands

security oauth2 modify

Modify global OAuth 2.0 configuration

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 modify` command enables or disables the OAuth 2.0 feature for token-based authentication.

Parameters

`[-enabled {true|false}]` - OAuth 2.0 Enabled

Use this parameter to enable or disable the OAuth 2.0 feature for the cluster.

Examples

The following example enables the OAuth 2.0 feature for the cluster:

```
cluster1::> security oauth2 modify -enabled true
```

security oauth2 show

Display global OAuth 2.0 configuration

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 show` command displays the status of the OAuth 2.0 feature.

Examples

The following example displays the OAuth 2.0 feature status information -

```
cluster1::> security oauth2 show
                     Is OAuth 2.0 Enabled: true
```

security oauth2 client create

Configure OAuth 2.0 Provider

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 client create` command creates OAuth 2.0 Provider configuration with the specified configuration name for token-based authentication. This command does not enable the OAuth 2.0 feature, it only configures it. Configuring and enabling the OAuth2.0 feature is a two-step process:

- Create an OAuth 2.0 Provider configuration using the `security oauth2 client create` command.
- Enable the OAuth 2.0 feature using the security oauth2 modify`-enabled` *true* command. This step must be performed once regardless of the number of providers configured. If you have already enabled the OAuth 2.0 feature as part of configuring another provider, you do not have to perform this step again for this provider.

After an OAuth 2.0 configuration is created, it cannot be modified. It must be deleted and created again to change any settings.

> Enabling/Disabling OAuth 2.0 restarts the web server. Any HTTP/S connections that are active will be disrupted.

Parameters

`-config-name <text>` - Configuration Entry Name

This is the OAuth 2.0 configuration entry name.

`-application <OAuth 2.0 Applications>` - Application

This is the application for which OAuth 2.0 is configured. Currently only the *http* application is supported.

`-issuer {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}` - OAuth 2.0 Issuer

This is the OAuth 2.0 issuer to match with the "iss" field from the access token.

`[-audience <text>]` - OAuth 2.0 Audience

This is the OAuth 2.0 audience to match with the "aud" field from the access token. If this parameter is not set, then the "aud" field will not be matched and the REST API request will be forwarded to the provider with the matching "iss" field.

`[-client-id <text>]` - OAuth 2.0 Client ID

This is the Client identifier used in token introspection calls to the IdP server.

`[-introspection-endpoint {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}]` - OAuth 2.0 Token Introspection Endpoint Location

This is the URI of the desired IdP server used for token introspection.

`[-introspection-interval {P[<integer>D]T[<integer>H][<integer>M][<integer>S] | P<integer>W | disabled}]` - OAuth 2.0 Token Introspection Refresh Interval in ISO-8601 format

This is the refresh interval in ISO-8601 format for caching the introspected tokens. When not set, the default value of *0s* is used, which caches introspected tokens for a period of "exp" value in the access token. This can be set to the value *disabled* to disable caching of tokens. Otherwise, it can be set to a value from *1s* to *2147483647s* .

`[-remote-user-claim <text>]` - OAuth 2.0 Remote User Claim

When the `-use-local-roles-if-present` parameter is set to true, and the token scope rules do not explicitly allow or deny the request, the value of the `-remote-user-claim` field will be used to find a match in the local user database for a user of the same name with the application of type *`http`* and authentication method *`password`* . When not set, the default value of *`sub`* is used.

`[-provider-jwks-uri {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}]` - OAuth 2.0 Provider JSON Web Key Set Location

This is the URI where the JSON Web Key Set (JWKS) is hosted by the Identity Provider server.

`[-jwks-refresh-interval {P[<integer>D]T[<integer>H][<integer>M][<integer>S] | P<integer>W}]` - OAuth 2.0 JSON Web Key Set Refresh Interval in ISO-8601 format

This is the refresh interval in ISO-8601 format for caching the JSON Web Key Set (JWKS) key set obtained from the provider-jwks-uri. When not set, the default value of *`1h`* is used. This can be set to a value from *`3600s`* to *`2147483647s`* .

`[-outgoing-proxy <text>]` - OAuth 2.0 Outgoing Proxy To Access External IdPs

This is the value for outgoing proxy to access external identity providers (IdPs). Use this parameter for local validation to download the JWKS key set in the JWKS URI, or for remote introspection for validating the access token in the Bearer field of the REST API request when the system is behind a proxy.

`[-use-local-roles-if-present {true|false}]` - Use Local Roles, If Present

When this parameter is set to *`true`* , and the scopes in the access token do not explicitly allow or deny the request, the local user role matching the user in the `-remote-user-claim` field (defaults to value of the "sub" field of the access token if not specified) will be checked for authorization of the request. The default value is *`false`* when not set, which means only the scopes in the access token is used to approve or deny the request.

`[-skip-uri-validation {true|false}]` - Skip URI Validations

When this parameter is set to *`true`* , validation of provider-jwks-uri is skipped. The default value of this parameter is *`false`* .

`[-use-mutual-tls {none|request|required}]` - Mutual TLS enforcement

This is the Mutual TLS setting for the OAuth 2.0 configuration. When set to *`required`* , OAuth 2.0 mutual TLS authentication is enforced for all access tokens and any token that does not have x5t#S256 property in the cnf section is rejected. The default value is *`request`* when not set, which means OAuth 2.0 mutual TLS authentication is enforced only if the x5t#S256 property is present in the cnf section of the access token. This can be disabled by setting to value *`none`* .

Examples

The following example creates OAuth 2.0 Provider configuration for Local Validation:

```
cluster1::> security oauth2 client create -config-name auth1 -application
http -issuer https://issuer.example.com/ -provider-jwks-uri
https://issuer.example.com/.well-known/jwks.json -use-local-roles-if
-present true -remote-user-claim preferred_username -outgoing-proxy
https://outgoing_proxy
```

The following example creates OAuth 2.0 Provider configuration for Remote Introspection:

```
cluster1::> security oauth2 client create -config-name auth1 -application
http -issuer https://issuer.example.com/ -client-id client_id -client
-secret client_secret -use-local-roles-if-present true -remote-user-claim
preferred_username -outgoing-proxy https://outgoing_proxy -use-mutual-tls
required
```

Related Links

- security oauth2 modify

security oauth2 client delete

Delete OAuth 2.0 Provider

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 client delete` command is used to remove an OAuth 2.0 Provider configuration.

Parameters

`-config-name <text>` - Configuration Entry Name

This is the OAuth 2.0 configuration entry name.

Examples

The following example removes the OAuth 2.0 Provider configuration named auth1:

```
cluster1::> security oauth2 client delete -config-name auth1
```

The following example removes all OAuth2.0 Provider configurations:

```
cluster1::> security oauth2 client delete -config-name *
```

security oauth2 client show

Display OAuth 2.0 Provider

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 client show` command displays the configured OAuth 2.0 Provider configuration.

Parameters

{ `[-fields <fieldname>,…]`

This specifies the fields that need to be displayed.

| `[-instance ]` }

If this parameter is specified, the command displays information about all OAuth 2.0 configuration entries.

`[-config-name <text>]` - Configuration Entry Name

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified config-name.

`[-application <OAuth 2.0 Applications>]` - Application

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified application. Currently only the *`http`* application is supported.

`[-issuer {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}]` - OAuth 2.0 Issuer

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified issuer.

`[-audience <text>]` - OAuth 2.0 Audience

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified audience.

`[-client-id <text>]` - OAuth 2.0 Client ID

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified client-id.

`[-hashed-client-secret <Hex String>]` - Hashed representation of client secret

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified hashed-client-secret.

`[-introspection-endpoint {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}]` - OAuth 2.0 Token Introspection Endpoint Location

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified introspection-endpoint.

`[-introspection-interval {P[<integer>D]T[<integer>H][<integer>M][<integer>S] | P<integer>W | disabled}]` - OAuth 2.0 Token Introspection Refresh Interval in ISO-8601 format

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified introspection-interval.

`[-remote-user-claim <text>]` - OAuth 2.0 Remote User Claim

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified remote-user-claim.

`[-provider-jwks-uri {scheme://(hostname|IPv4 Address|'['IPv6 Address']')…}]` - OAuth 2.0 Provider JSON Web Key Set Location

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified provider-jwks-uri.

`[-jwks-refresh-interval {P[<integer>D]T[<integer>H][<integer>M][<integer>S] | P<integer>W}]` - OAuth 2.0 JSON Web Key Set Refresh Interval in ISO-8601 format

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified jwks-refresh-interval.

`[-outgoing-proxy <text>]` - OAuth 2.0 Outgoing Proxy To Access External IdPs

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified outgoing-proxy.

`[-use-local-roles-if-present {true|false}]` - Use Local Roles, If Present

If this parameter is specified, the command displays information only about the OAuth 2.0 configuration that match the specified use-local-roles-if-present.

`[-use-mutual-tls {none|request|required}]` - Mutual TLS enforcement

This is the Mutual TLS setting for the OAuth 2.0 configuration. When set to *`required`* , OAuth 2.0 mutual TLS authentication is enforced for all access tokens and any token that does not have x5t#S256 property in the cnf section is rejected. The default value is *`request`* when not set, which means OAuth 2.0 mutual TLS authentication is enforced only if the x5t#S256 property is present in the cnf section of the access token. This can be disabled by setting to value *`none`* .

Examples

The following example displays the OAuth 2.0 Provider configuration for Local Validation:

```
cluster1::> security oidc client show
                                Configuration Name: auth1
                                       Application: http
                          Issuer: https://issuer.example.com/
                                          Audience: -
                                         Client ID: -
                              Hashed Client Secret: -
                            Introspection Endpoint: -
                   Introspection Refresh Interval : -
                                   Use local roles: true
                Provider JSON Web Key Set Location:
https://issuer.example.com/.well-known/jwks.json
                 JSON Web Key Set Refresh Interval: 1h
                                 Remote User Claim: preferred_username
                                    Outgoing Proxy: https://outgoing_proxy
                            Mutual TLS enforcement: request
```

The following example displays the OAuth 2.0 Provider configuration for Remote Introspection:

```
cluster1::> security oidc client show
                                Configuration Name: auth1
                                       Application: http
                                            Issuer:
https://issuer.example.com/
                                          Audience: -
                                         Client ID: client_id
                              Hashed Client Secret:
e194e3472ee55c4202582cfbf59a03a37ef27085d2baf1b2fd7f7da3973c56fa
                            Introspection Endpoint: -
                   Introspection Refresh Interval : 0s
                                   Use local roles: true
                Provider JSON Web Key Set Location: -
                 JSON Web Key Set Refresh Interval: -
                                 Remote User Claim: preferred_username
                                    Outgoing Proxy: https://outgoing_proxy
                            Mutual TLS enforcement: required
```

security oauth2 scope cli-to-scope generate

Generate OAuth 2.0 scope for the given CLI REST role creation command parameters

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 scope cli-to-scope generate` command generates on ONTAP-specific OAuth 2.0 scope string based on local ONTAP custom roles created using security login rest-role create.

Parameters

`-role <text>` - Role name

The role name as in the security login rest-role create -role parameter. This parameter is required.

`-access <text>` - Access level

The access level as in the security login rest-role create -access parameter. Valid access levels are none, readonly, all, read_create, read_modify and read_create_modify. This parameter is required.

`[-api <text>]` - API path

The REST API URI as in the security login rest-role create -api parameter. Valid APIs start with /api/. This parameter is required.

`[-cluster-uuid <text>]` - Cluster UUID

The cluster UUID for which this scope applies. This parameter is optional. If not specified, the OAuth 2.0 scope is applicable to all clusters

Examples

To generate the OAuth 2.0 scope string applicable to all clusters for an ONTAP role named myrole for the REST API URI /api/cluster with admin (all) access:

```
cluster1::gt; security oauth2 scope cli-to-scope generate -role myrole
-api /api/cluster -access all -cluster-uuid *
ontap:*:myrole:all:*:/api/cluster
```

security oauth2 scope scope-to-cli generate

Generate CLI REST role command for the given OAuth 2.0 scope

Availability: This command is available to *cluster* administrators at the *admin* privilege level.

Description

The `security oauth2 scope scope-to-cli generate` command generates on ONTAP CLI security login rest-role create command that is the equivalent of the specified OAuth 2.0 scope string.

Parameters

`-scopeString <text>` - OAuth 2.0 scope

The OAuth 2.0 scope string. This parameter is required.

Examples

To generate ONTAP CLI command given an OAuth 2.0 scope string applicable to all clusters for an ONTAP role named restclusterrole for the REST API URI /api/cluster with readonly access:

```
cluster1::gt; security oauth2 scope scope-to-cli generate -scopeString
ontap:*:restclusterrole:readonly:*:/api/cluster
Command for cluster <All>:
security login rest-role create -role restclusterrole -access readonly
-api /api/cluster
```

Copyright information

Trademark information