+++ title = "Fine-grained access control HTTP API " description = "Fine-grained access control API" keywords = ["grafana", "http", "documentation", "api", "fine-grained-access-control", "acl", "enterprise"] aliases = ["/docs/grafana/latest/http_api/accesscontrol/"] +++

Fine-grained access control API

> *Fine-grained access control API is only available in Grafana Enterprise. Read more about [Grafana Enterprise]({{< relref "../enterprise" >}}).*

The API can be used to create, update, get and list roles, and create or remove built-in role assignments. To use the API, you would need to [enable fine-grained access control]({{< relref "../enterprise/access-control/_index.md#enable-fine-grained-access-control" >}}).

The API does not currently work with an API Token. So in order to use these API endpoints you will have to use [Basic auth]({{< relref "./auth/#basic-auth" >}}).

Get status

`GET /api/access-control/status`

Returns an indicator to check if fine-grained access control is enabled or not.

Required permissions

Action	Scope
status:accesscontrol	services:accesscontrol

Example request

```
GET /api/access-control/status
Accept: application/json
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
  "enabled": true
}
```

Status codes

Code	Description
200	Returned a flag indicating if the fine-grained access control is enabled or no.
403	Access denied

404	Not found, an indication that fine-grained access control is not available at all.
500	Unexpected error. Refer to body and/or server logs for more details.

Create and manage custom roles

Get all roles

`GET /api/access-control/roles`

Gets all existing roles. The response contains all global and organization local roles, for the organization which user is signed in.

Refer to the [Role scopes]({{< relref "../enterprise/access-control/roles.md#built-in-role-assignments" >}}) for more information.

Query Parameters:

- `includeHidden` : Optional. Set to `true` to include roles that are `hidden` .

Required permissions

Action	Scope
roles:list	roles:*

Example request

```
GET /api/access-control/roles
Accept: application/json
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

[
    {
        "version": 3,
        "uid": "XvHQJq57z",
        "name": "fixed:reports:reader",
        "displayName": "Report reader",
        "description": "Read all reports and shared report settings.",
        "group": "Reports",
        "updated": "2021-11-19T10:48:00+01:00",
        "created": "2021-11-19T10:48:00+01:00",
        "global": false
    },
    {
        "version": 5,
        "uid": "vi9mlLjGz",
        "name": "fixed:datasources.permissions:writer",
        "description: "Create, read or delete data source permissions.",
        "global": true,
        "updated": "2021-05-13T22:41:49+02:00",
        "created": "2021-05-13T16:24:26+02:00"
    }
]
```

Status codes

Code	Description
200	Global and organization local roles are returned.
403	Access denied
500	Unexpected error. Refer to body and/or server logs for more details.

Get a role

`GET /api/access-control/roles/:uid`

Get a role for the given UID.

Required permissions

Action	Scope
roles:read	roles:*

Example request

```
GET /api/access-control/roles/PYnDO3rMk
Accept: application/json
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "version": 4,
    "uid": "6dNwJq57z",
    "name": "fixed:reports:writer",
    "displayName": "Report writer",
    "description": "Create, read, update, or delete all reports and shared report
settings.",
    "group": "Reports",
    "permissions": [
        {
            "action": "reports:delete",
            "scope": "reports:*",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports:read",
            "scope": "reports:*",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports:send",
            "scope": "reports:*",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports.admin:create",
            "scope": "",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports.admin:write",
            "scope": "reports:*",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports.settings:read",
            "scope": "",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        },
        {
            "action": "reports.settings:write",
            "scope": "",
            "updated": "2021-11-19T10:48:00+01:00",
            "created": "2021-11-19T10:48:00+01:00"
        }
    ],
    "updated": "2021-11-19T10:48:00+01:00",
    "created": "2021-11-19T10:48:00+01:00",
    "global": false
}
```

Status codes

Code	Description
200	Role is returned.

403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

Create a new custom role

`POST /api/access-control/roles`

Creates a new custom role and maps given permissions to that role. Note that roles with the same prefix as [Fixed Roles]({{< relref "../enterprise/access-control/roles.md" >}}) can't be created.

Required permissions

`permission:delegate` scope ensures that users can only create custom roles with the same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to create a custom role which allows to do that. This is done to prevent escalation of privileges.

Action	Scope
roles:write	permissions:delegate

Example request

```
POST /api/access-control/roles
Accept: application/json
Content-Type: application/json

{
    "version": 1,
    "uid": "jZrmlLCGka",
    "name": "custom:delete:roles",
    "displayName": "custom delete roles",
    "description": "My custom role which gives users permissions to delete roles",
    "group":"My Group",
    "displayName": "My Custom Role",
    "global": false,
    "permissions": [
        {
            "action": "roles:delete",
            "scope": "permissions:delegate"
        }
    ]
}
```

JSON body schema

Field Name	Date Type	Required	Description
uid	string	No	UID of the role. If not present, the UID will be automatically created for you and returned in response. Refer to the [Custom roles]({{< relref "../enterprise/access-control/roles.md#custom-roles" >}}) for more information.

global	boolean	No	A flag indicating if the role is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request. Refer to the [Role scopes]({{< relref "../enterprise/access-control/roles.md#role-scopes" >}}) for more information.
version	number	No	Version of the role. If not present, version 0 will be assigned to the role and returned in the response. Refer to the [Custom roles]({{< relref "../enterprise/access-control/roles.md#custom-roles" >}}) for more information.
name	string	Yes	Name of the role. Refer to [Custom roles]({{< relref "../enterprise/access-control/roles.md#custom-roles" >}}) for more information.
description	string	No	Description of the role.
displayName	string	No	Display name of the role, visible in the UI.
group	string	No	The group name the role belongs to.
hidden	boolean	No	Specify whether the role is hidden or not. If set to `true`, then the role does not show in the role picker. It will not be listed by API endpoints unless explicitly specified.
permissions	Permission	No	If not present, the role will be created without any permissions.

Permission

Field Name	Data Type	Required	Description
action	string	Yes	Refer to [Permissions]({{< relref "../enterprise/access-control/permissions.md" >}}) for full list of available actions.
scope	string	No	If not present, no scope will be mapped to the permission. Refer to [Permissions]({{< relref "../enterprise/access-control/permissions.md#scope-definitions" >}}) for full list of available scopes.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "version": 2,
    "uid": "jZrmlLCGka",
    "name": "custom:delete:create:roles",
    "displayName": "custom delete create roles",
    "description": "My custom role which gives users permissions to delete and create roles",
    "group":"My Group",
    "displayName": "My Custom Role",
    "global": false,
    "permissions": [
        {
            "action": "roles:delete",
            "scope": "permissions:delegate",
            "updated": "2021-05-13T23:19:46+02:00",
            "created": "2021-05-13T23:19:46+02:00"
        }
    ],
    "updated": "2021-05-13T23:20:51.416518+02:00",
    "created": "2021-05-13T23:19:46+02:00"
}
```

Status codes

Code	Description
200	Role is updated.
400	Bad request (invalid json, missing content-type, missing or invalid fields, etc.).
403	Access denied
500	Unexpected error. Refer to body and/or server logs for more details.

Update a custom role

`PUT /api/access-control/roles/:uid`

Update the role with the given UID, and it's permissions with the given UID. The operation is idempotent and all permissions of the role will be replaced with what is in the request. You would need to increment the version of the role with each update, otherwise the request will fail.

Required permissions

`permission:delegate` scope ensures that users can only update custom roles with the same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to update a custom role which allows to do that. This is done to prevent escalation of privileges.

Action	Scope
roles:write	permissions:delegate

Example request

```
PUT /api/access-control/roles/jZrmlLCGka
Accept: application/json
Content-Type: application/json

{
    "version": 3,
    "name": "custom:delete:write:roles",
    "displayName": "custom delete write roles",
    "description": "My custom role which gives users permissions to delete and write roles",
    "group":"My Group",
    "displayName": "My Custom Role",
    "global": false,
    "permissions": [
        {
            "action": "roles:delete",
            "scope": "permissions:delegate"
        },
        {
            "action": "roles:write",
            "scope": "permissions:delegate"
        }
    ]
}
```

JSON body schema

Field Name	Data Type	Required	Description
version	number	Yes	Version of the role. Must be incremented for update to work.
name	string	Yes	Name of the role.
description	string	No	Description of the role.
displayName	string	No	Display name of the role, visible in the UI.
group	string	No	The group name the role belongs to.
hidden	boolean	No	Specify whether the role is hidden or not. If set to `true`, then the role does not show in the role picker. It will not be listed by API endpoints unless explicitly specified.
permissions	List of Permissions	No	The full list of permissions for the role after the update.

Permission

Field Name	Data Type	Required	Description
action	string	Yes	Refer to [Permissions]({{< relref "../enterprise/access-control/permissions.md" >}}) for full list of available actions.
scope	string	No	If not present, no scope will be mapped to the permission. Refer to [Permissions]({{< relref "../enterprise/access-control/permissions.md#scope-definitions" >}}) for full list of available scopes.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
```

```
{
    "version":3,
    "uid":"jZrmlLCGka",
    "name":"custom:delete:write:roles",
    "displayName":"custom delete write roles",
    "description":"My custom role which gives users permissions to delete and write roles",
    "group":"My Group",
    "displayName": "My Custom Role",
    "permissions":[
        {
            "action":"roles:delete",
            "scope":"permissions:delegate",
            "updated":"2021-08-06T18:27:40+02:00",
            "created":"2021-08-06T18:27:40+02:00"
        },
        {
            "action":"roles:write",
            "scope":"permissions:delegate",
            "updated":"2021-08-06T18:27:41+02:00",
            "created":"2021-08-06T18:27:41+02:00"
        }
    ],
    "updated":"2021-08-06T18:27:41+02:00",
    "created":"2021-08-06T18:27:40+02:00",
    "global":false
}
```

Status codes

Code	Description
200	Role is updated.
400	Bad request (invalid json, missing content-type, missing or invalid fields, etc.).
403	Access denied
404	Role was not found to update.
500	Unexpected error. Refer to body and/or server logs for more details.

Delete a custom role

`DELETE /api/access-control/roles/:uid?force=false`

Delete a role with the given UID, and it's permissions. If the role is assigned to a built-in role, the deletion operation will fail, unless `force` query param is set to `true`, and in that case all assignments will also be deleted.

Required permissions

`permission:delegate` scope ensures that users can only delete a custom role with the same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to delete a custom role which allows to do that.

Action	Scope
roles:delete	permissions:delegate

Example request

```
DELETE /api/access-control/roles/jZrmlLCGka?force=true&global=false
Accept: application/json
```

Query parameters

Param	Type	Required	Description
force	boolean	No	When set to `true`, the role will be deleted with all it's assignments.
global	boolean	No	A flag indicating if the role is global or not. If set to false, the default org ID of the authenticated user will be used from the request. Refer to the [Role scopes]({{< relref "../enterprise/access-control/roles.md#built-in-role-assignments" >}}) for more information.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Role deleted"
}
```

Status codes

Code	Description
200	Role is deleted.
400	Bad request (invalid json, missing content-type, missing or invalid fields, etc.).
403	Access denied
500	Unexpected error. Refer to body and/or server logs for more details.

Create and remove user role assignments

List roles assigned to a user

`GET /api/access-control/users/:userId/roles`

Lists the roles that have been directly assigned to a given user. The list does not include built-in roles (Viewer, Editor, Admin or Grafana Admin), and it does not include roles that have been inherited from a team.

Query Parameters:

- `includeHidden` : Optional. Set to `true` to include roles that are `hidden` .

Required permissions

Action	Scope
users.roles:list	users:id:`<user ID>`

Example request

```
GET /api/access-control/users/1/roles
Accept: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

[
    {
        "version": 4,
        "uid": "6dNwJq57z",
        "name": "fixed:reports:writer",
        "displayName": "Report writer",
        "description": "Create, read, update, or delete all reports and shared
report settings.",
        "group": "Reports",
        "updated": "2021-11-19T10:48:00+01:00",
        "created": "2021-11-19T10:48:00+01:00",
        "global": false
    }
]
```

Status codes

Code	Description
200	Set of assigned roles is returned.
403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

List permissions assigned to a user

`GET /api/access-control/users/:userId/permissions`

Lists the permissions that a given user has.

Required permissions

Action	Scope

users.permissions:list	users:id:`<user ID>`

Example request

```
GET /api/access-control/users/1/permissions
Accept: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

[
    {
        "action": "ldap.status:read",
        "scope": ""
    },
    {
        "action": "ldap.user:read",
        "scope": ""
    }
]
```

Status codes

Code	Description
200	Set of assigned permissions is returned.
403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

Add a user role assignment

`POST /api/access-control/users/:userId/roles`

Assign a role to a specific user.

For bulk updates consider [Set user role assignments]({{< ref "#set-user-role-assignments" >}}).

Required permissions

`permission:delegate` scope ensures that users can only assign roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to assign a role which will allow to do that. This is done to prevent escalation of privileges.

Action	Scope
users.roles:add	permissions:delegate

Example request

```
POST /api/access-control/users/1/roles
Accept: application/json
Content-Type: application/json

{
    "global": false,
    "roleUid": "XvHQJq57z"
}
```

JSON body schema

Field Name	Data Type	Required	Description
roleUid	string	Yes	UID of the role.
global	boolean	No	A flag indicating if the assignment is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request to create organization local assignment.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Role added to the user."
}
```

Status codes

Code	Description
200	Role is assigned to a user.
403	Access denied.
404	Role not found.
500	Unexpected error. Refer to body and/or server logs for more details.

Remove a user role assignment

`DELETE /api/access-control/users/:userId/roles/:roleUID`

Revoke a role from a user.

For bulk updates consider [Set user role assignments]({{< ref "#set-user-role-assignments" >}}).

Required permissions

`permission:delegate` scope ensures that users can only unassign roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to unassign a role which will allow to do that. This is done to prevent escalation of privileges.

Action	Scope
users.roles:remove	permissions:delegate

Query parameters

Param	Type	Required	Description
global	boolean	No	A flag indicating if the assignment is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request to remove assignment.

Example request

```
DELETE /api/access-control/users/1/roles/AFUXBHKnk
Accept: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Role removed from user."
}
```

Status codes

Code	Description
200	Role is unassigned.
403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

Set user role assignments

`PUT /api/access-control/users/:userId/roles`

Update the user's role assignments to match the provided set of UIDs. This will remove any assigned roles that aren't in the request and add roles that are in the set but are not already assigned to the user.

If you want to add or remove a single role, consider using [Add a user role assignment]({{< ref "#add-a-user-role-assignment" >}}) or [Remove a user role assignment]({{< ref "#remove-a-user-role-assignment" >}}) instead.

Required permissions

`permission:delegate` scope ensures that users can only assign or unassign roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to assign or unassign a role which will allow to do that. This is done to prevent escalation of privileges.

Action	Scope
users.roles:add	permissions:delegate
users.roles:remove	permissions:delegate

Example request

```
PUT /api/access-control/users/1/roles
Accept: application/json
Content-Type: application/json

{
    "global": false,
    "roleUids": [
        "ZiHQJq5nk",
        "GzNQ1357k"
    ]
}
```

JSON body schema

Field Name	Date Type	Required	Description
global	boolean	No	A flag indicating if the assignment is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request.
roleUids	list	Yes	List of role UIDs.
includeHidden	boolean	No	Specify whether the hidden role assignments should be updated.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "User roles have been updated."
}
```

Status codes

Code	Description
200	Roles have been assigned.
403	Access denied.
404	Role not found.
500	Unexpected error. Refer to body and/or server logs for more details.

Create and remove team role assignments

List roles assigned to a team

`GET /api/access-control/teams/:teamId/roles`

Lists the roles that have been directly assigned to a given team.

Query Parameters:

- `includeHidden` : Optional. Set to `true` to include roles that are `hidden` .

Required permissions

Action	Scope
teams.roles:list	teams:id:`<team ID>`

Example request

```
GET /api/access-control/teams/1/roles
Accept: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

[
    {
        "version": 4,
        "uid": "j08ZBi-nk",
        "name": "fixed:licensing:reader",
        "displayName": "Licensing reader",
        "description": "Read licensing information and licensing reports.",
        "group": "Licenses",
        "updated": "2022-02-03T14:19:50+01:00",
        "created": "0001-01-01T00:00:00Z",
        "global": false
    }
]
```

Status codes

Code	Description
200	Set of assigned roles is returned.
403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

Add a team role assignment

`POST /api/access-control/teams/:teamId/roles`

Assign a role to a specific team.

For bulk updates consider [Set team role assignments]({{< ref "#set-team-role-assignments" >}}).

Required permissions

`permission:delegate` scope ensures that users can only assign roles which have same, or a subset of permissions which the user has. For example, if a user does not have the permissions required to create users, they won't be able to assign a role that contains these permissions. This is done to prevent escalation of privileges.

Action	Scope
teams.roles:add	permissions:delegate

Example request

```
POST /api/access-control/teams/1/roles
Accept: application/json
Content-Type: application/json

{
    "roleUid": "XvHQJq57z"
}
```

JSON body schema

Field Name	Data Type	Required	Description
roleUid	string	Yes	UID of the role.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Role added to the team."
}
```

Status codes

Code	Description
200	Role is assigned to a team.
403	Access denied.
404	Role not found.

500	Unexpected error. Refer to body and/or server logs for more details.

Remove a team role assignment

`DELETE /api/access-control/teams/:teams/roles/:roleUID`

Revoke a role from a team.

For bulk updates consider [Set team role assignments]({{< ref "#set-team-role-assignments" >}}).

Required permissions

`permission:delegate` scope ensures that users can only unassign roles which have same, or a subset of permissions which the user has. For example, if a user does not have the permissions required to create users, they won't be able to assign a role that contains these permissions. This is done to prevent escalation of privileges.```

Action	Scope
teams.roles:remove	permissions:delegate

Example request

```
DELETE /api/access-control/teams/1/roles/AFUXBHKnk
Accept: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Role removed from team."
}
```

Status codes

Code	Description
200	Role is unassigned.
403	Access denied.
500	Unexpected error. Refer to body and/or server logs for more details.

Set team role assignments

`PUT /api/access-control/teams/:teamId/roles`

Update the team's role assignments to match the provided set of UIDs. This will remove any assigned roles that aren't in the request and add roles that are in the set but are not already assigned to the user.

If you want to add or remove a single role, consider using [Add a team role assignment]({{< ref "#add-a-team-role-assignment" >}}) or [Remove a team role assignment]({{< ref "#remove-a-team-role-assignment" >}}) instead.

Required permissions

`permission:delegate` scope ensures that users can only assign or unassign roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to assign or unassign a role to a team which will allow to do that. This is done to prevent escalation of privileges.

Action	Scope
teams.roles:add	permissions:delegate
teams.roles:remove	permissions:delegate

Example request

```
PUT /api/access-control/teams/1/roles
Accept: application/json
Content-Type: application/json

{
    "roleUids": [
        "ZiHQJq5nk",
        "GzNQ1357k"
    ]
}
```

JSON body schema

Field Name	Date Type	Required	Description
roleUids	list	Yes	List of role UIDs.
includeHidden	boolean	No	Specify whether the hidden role assignments should be updated.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Team roles have been updated."
}
```

Status codes

Code	Description

200	Roles have been assigned.
403	Access denied.
404	Role not found.
500	Unexpected error. Refer to body and/or server logs for more details.

Create and remove built-in role assignments

API set allows to create or remove [built-in role assignments]({{< relref "../enterprise/access-control/roles.md#built-in-role-assignments" >}}) and list current assignments.

Get all built-in role assignments

`GET /api/access-control/builtin-roles`

Gets all built-in role assignments.

Query Parameters:

- `includeHidden` : Optional. Set to `true` to include roles that are `hidden` .

Required permissions

Action	Scope
roles.builtin:list	roles:*

Example request

```
GET /api/access-control/builtin-roles
Accept: application/json
Content-Type: application/json
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "Admin": [
        {
            "version": 1,
            "uid": "qQui_LCMk",
            "name": "fixed:users:writer",
            "name": "User writer",
            "description": "Read and update all attributes and settings for all
users in Grafana: update user information, read user information, create or enable
or disable a user, make a user a Grafana administrator, sign out a user, update a
user’s authentication token, or update quotas for all users",
            "global": true,
            "updated": "2021-05-13T16:24:26+02:00",
            "created": "2021-05-13T16:24:26+02:00"
        },
        {
            "version": 1,
            "uid": "PeXmlYjMk",
            "name": "fixed:users:reader",
            "displayName": "User reader",
            "description": "Allows every read action for user organizations and in
addition allows to administer user organizations",
            "global": true,
            "updated": "2021-05-13T16:24:26+02:00",
            "created": "2021-05-13T16:24:26+02:00"
        }
    ],
    "Grafana Admin": [
        {
            "version": 1,
            "uid": "qQui_LCMk",
            "name": "fixed:users:writer",
            "displayName": "User writer",
            "description": "Read and update all attributes and settings for all
users in Grafana: update user information, read user information, create or enable
or disable a user, make a user a Grafana administrator, sign out a user, update a
user’s authentication token, or update quotas for all users",
            "global": true,
            "updated": "2021-05-13T16:24:26+02:00",
            "created": "2021-05-13T16:24:26+02:00"
        }
    ]
}
```

Status codes

Code	Description
200	Built-in role assignments are returned.
403	Access denied
500	Unexpected error. Refer to body and/or server logs for more details.

Create a built-in role assignment

`POST /api/access-control/builtin-roles`

Creates a new built-in role assignment.

Required permissions

`permission:delegate` scope ensures that users can only create built-in role assignments with the roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to create a built-in role assignment which will allow to do that. This is done to prevent escalation of privileges.

Action	Scope
roles.builtin:add	permissions:delegate

Example request

```
POST /api/access-control/builtin-roles
Accept: application/json
Content-Type: application/json

{
    "roleUid": "LPMGN99Mk",
    "builtinRole": "Grafana Admin",
    "global": false
}
```

JSON body schema

Field Name	Date Type	Required	Description
roleUid	string	Yes	UID of the role.
builtinRole	boolean	Yes	Can be one of `Viewer`, `Editor`, `Admin` or `Grafana Admin`.
global	boolean	No	A flag indicating if the assignment is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request to create organization local assignment. Refer to the [Built-in role assignments]({{< relref "../enterprise/access-control/roles.md#built-in-role-assignments" >}}) for more information.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Built-in role grant added"
}
```

Status codes

Code	Description
200	Role was assigned to built-in role.
400	Bad request (invalid json, missing content-type, missing or invalid fields, etc.).
403	Access denied
404	Role not found
500	Unexpected error. Refer to body and/or server logs for more details.

Remove a built-in role assignment

`DELETE /api/access-control/builtin-roles/:builtinRole/roles/:roleUID`

Deletes a built-in role assignment (for one of *Viewer*, *Editor*, *Admin*, or *Grafana Admin*) to the role with the provided UID.

Required permissions

`permission:delegate` scope ensures that users can only remove built-in role assignments with the roles which have same, or a subset of permissions which the user has. For example, if a user does not have required permissions for creating users, they won't be able to remove a built-in role assignment which allows to do that.

Action	Scope
roles.builtin:remove	permissions:delegate

Example request

```
DELETE /api/access-control/builtin-roles/Grafana%20Admin/roles/LPMGN99Mk?
global=false
Accept: application/json
```

Query parameters

Param	Type	Required	Description
global	boolean	No	A flag indicating if the assignment is global or not. If set to `false`, the default org ID of the authenticated user will be used from the request to remove assignment. Refer to the [Built-in role assignments]({{< relref "../enterprise/access-control/roles.md#built-in-role-assignments" >}}) for more information.

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8

{
    "message": "Built-in role grant removed"
}
```

Status codes

Code	Description
200	Role was unassigned from built-in role.
400	Bad request (invalid json, missing content-type, missing or invalid fields, etc.).
403	Access denied

404	Role not found.
500	Unexpected error. Refer to body and/or server logs for more details.