+++ title = “Provisioning roles and assignments” description = “Understand how to provision roles and assignments in fine-grained access control” keywords = [“grafana”, “fine-grained-access-control”, “roles”, “provisioning”, “assignments”, “permissions”, “enterprise”] weight = 120 +++

Provisioning

You can create, change or remove [Custom roles]({{< relref “./roles.md#custom-roles” >}}) and create or remove built-in role assignments({{< relref “./roles.md#built-in-role-assignments” >}}), by adding one or more YAML configuration files in the [`provisioning/access-control/`]({{< relref “../../administration/configuration/#provisioning” >}}) directory. Refer to [Grafana provisioning]({{< relref “../../administration/configuration/#provisioning” >}}) to learn more about provisioning.

If you want to manage roles and built-in role assignments by API, refer to the [Fine-grained access control HTTP API]({{< relref “../../http_api/access_control/” >}}).

Configuration

The configuration files must be placed in [`provisioning/access-control/`]({{< relref “../../administration/configuration/#provisioning” >}}). Grafana performs provisioning during the startup. Refer to the [Reload provisioning configurations]({{< relref “../../http_api/admin/#reload-provisioning-configurations” >}}) to understand how you can reload configuration at runtime.

Manage custom roles

You can create, update, and delete custom roles, as well as create and remove built-in role assignments.

Create or update roles

To create or update custom roles, you can add a list of `roles` in the configuration.

Every role has a [version]({{< relref “./roles.md#custom-roles” >}}) number. For each role you update, you must remember to increment it, otherwise changes won’t be applied.

When you update a role, the existing role inside Grafana is altered to be exactly what is specified in the YAML file, including permissions.

Here is an example YAML file to create a local role with a set of permissions:

```
# config file version
apiVersion: 1
```

```yaml
# Roles to insert into the database, or roles to update in the database
roles:
  - name: custom:users:editor
    description: 'This role allows users to list, create, or update other users within the o
    version: 1
    orgId: 1
    permissions:
      - action: 'users:read'
        scope: 'users:*'
      - action: 'users:write'
        scope: 'users:*'
      - action: 'users:create'
        scope: 'users:*'
```

Here is an example YAML file to create a hidden global role with a set of permissions. `global:true` option makes a role global, and `hidden:true` option hides the role from the role picker:

```yaml
# config file version
apiVersion: 1

# Roles to insert into the database, or roles to update in the database
roles:
  - name: custom:users:editor
    description: 'This role allows users to list, create, or update other users within the o
    version: 1
    global: true
    hidden: true
    permissions:
      - action: 'users:read'
        scope: 'users:*'
      - action: 'users:write'
        scope: 'users:*'
      - action: 'users:create'
        scope: 'users:*'
```

The `orgId` is lost when the role is set to global.

Delete roles

To delete a role, add a list of roles under the `deleteRoles` section in the configuration file.

> **Note:** Any role in the `deleteRoles` section is deleted before any role in the `roles` section is saved.

Here is an example YAML file to delete a role:

```yaml
# config file version
```

```yaml
apiVersion: 1

# list of roles that should be deleted
deleteRoles:
  - name: custom:reports:editor
    orgId: 1
    force: true
```

Assign your custom role to specific built-in roles

To assign roles to built-in roles, add said built-in roles to the `builtInRoles` section of your roles. To remove a specific assignment, remove it from the list.

> **Note:** Assignments are updated if the version of the role is greater or equal to the one stored internally. You don't need to increment the version number of the role to update its assignments.

For example, the following role is assigned to an organization editor or an organization administrator:

```yaml
# config file version
apiVersion: 1

# Roles to insert/update in the database
roles:
  - name: custom:users:editor
    description: 'This role allows users to list/create/update other users in the organizati
    version: 1
    orgId: 1
    permissions:
      - action: 'users:read'
        scope: 'users:*'
      - action: 'users:write'
        scope: 'users:*'
      - action: 'users:create'
        scope: 'users:*'
    builtInRoles:
      - name: 'Editor'
      - name: 'Admin'
```

Assign your custom role to specific teams

To assign roles to teams, add said teams to the `teams` section of your roles. To remove a specific assignment, remove it from the list.

> **Note:** Assignments are updated if the version of the role is greater or equal to the one stored internally. You don't need to increment the version number of the role to update its assignments. Assignments

to built-in roles will be ignored. Use `addDefaultAssignments` and `removeDefaultAssignments` instead.

In order for provisioning to succeed, specified teams must already exist. Additionally, since teams are local to an organization, the organization has to be specified in the assignment.

For example, the following role is assigned to the `user editors` team and `user admins` team:

```yaml
# config file version
apiVersion: 1

# Roles to insert/update in the database
roles:
  - name: custom:users:writer
    description: 'List/update other users in the organization'
    version: 1
    global: true
    permissions:
      - action: 'org.users:read'
        scope: 'users:*'
      - action: 'org.users:write'
        scope: 'users:*'
    teams:
      - name: 'user editors'
        orgId: 1
      - name: 'user admins'
        orgId: 1
```

Assign fixed roles to specific teams

To assign a fixed role to teams, add said teams to the `teams` section of the associated entry. To remove a specific assignment, remove it from the list.

> **Note:** Since fixed roles are global, the Global attribute has to be specified. A fixed role will never be updated through provisioning.

In order for provisioning to succeed, specified teams must already exist. Additionally, since teams are local to an organization, the organization has to be specified in the assignment.

For example, the following fixed role is assigned to the `user editors` team and `user admins` team:

```yaml
# config file version
apiVersion: 1

# Roles to insert/update in the database
```

```yaml
roles:
  - name: fixed:users:writer
    global: true
    teams:
      - name: 'user editors'
        orgId: 1
      - name: 'user admins'
        orgId: 1
```

Manage default built-in role assignments

During startup, Grafana creates [default built-in role assignments]({{< relref "./roles#default-built-in-role-assignments" >}}) with [fixed roles]({{< relref "./roles#fixed-roles" >}}). You can remove and later restore those assignments with provisioning.

Remove default assignment

To remove default built-in role assignments, use the `removeDefaultAssignments` element in the configuration file. You need to provide the built-in role name and fixed role name.

Here is an example:

```yaml
# config file version
apiVersion: 1

# list of default built-in role assignments that should be removed
removeDefaultAssignments:
  - builtInRole: 'Grafana Admin'
    fixedRole: 'fixed:permissions:admin'
```

Restore default assignment

To restore the default built-in role assignment, use the `addDefaultAssignments` element in the configuration file. You need to provide the built-in role name and the fixed-role name.

Here is an example:

```yaml
# config file version
apiVersion: 1

# list of default built-in role assignments that should be added back
addDefaultAssignments:
  - builtInRole: 'Admin'
    fixedRole: 'fixed:reporting:admin:read'
```

Full example of a role configuration file

```yaml
# config file version
apiVersion: 1

# list of default built-in role assignments that should be removed
removeDefaultAssignments:
  # <string>, must be one of the Organization roles (`Viewer`, `Editor`, `Admin`) or `Grafa
  - builtInRole: 'Grafana Admin'
    # <string>, must be one of the existing fixed roles
    fixedRole: 'fixed:permissions:admin'

# list of default built-in role assignments that should be added back
addDefaultAssignments:
  # <string>, must be one of the Organization roles (`Viewer`, `Editor`, `Admin`) or `Grafa
  - builtInRole: 'Admin'
    # <string>, must be one of the existing fixed roles
    fixedRole: 'fixed:reporting:admin:read'

# list of roles that should be deleted
deleteRoles:
  # <string> name of the role you want to create. Required if no uid is set
  - name: 'custom:reports:editor'
    # <string> uid of the role. Required if no name
    uid: 'customreportseditor1'
    # <int> org id. will default to Grafana's default if not specified
    orgId: 1
    # <bool> force deletion revoking all grants of the role
    force: true
  - name: 'custom:global:reports:reader'
    uid: 'customglobalreportsreader1'
    # <bool> overwrite org id and removes a global role
    global: true
    force: true

# list of roles to insert/update depending on what is available in the database
roles:
  # <string, required> name of the role you want to create. Required
  - name: 'custom:users:editor'
    # <string> uid of the role. Has to be unique for all orgs.
    uid: customuserseditor1
    # <string> description of the role, informative purpose only.
    description: 'Role for our custom user editors'
    # <int> version of the role, Grafana will update the role when increased
    version: 2
    # <int> org id. will default to Grafana's default if not specified
```

```
    orgId: 1
    # <list> list of the permissions granted by this role
    permissions:
      # <string, required> action allowed
      - action: 'users:read'
        #<string> scope it applies to
        scope: 'users:*'
      - action: 'users:write'
        scope: 'users:*'
      - action: 'users:create'
        scope: 'users:*'
    # <list> list of builtIn roles the role should be assigned to
    builtInRoles:
      # <string, required> name of the builtin role you want to assign the role to
      - name: 'Editor'
        # <int> org id. will default to the role org id
        orgId: 1
  - name: 'custom:global:users:reader'
    uid: 'customglobalusersreader1'
    description: 'Global Role for custom user readers'
    version: 1
    # <bool> overwrite org id and creates a global role
    global: true
    permissions:
      - action: 'users:read'
        scope: 'users:*'
    builtInRoles:
      - name: 'Viewer'
        orgId: 1
      - name: 'Editor'
        # <bool> overwrite org id and assign role globally
        global: true
  - name: fixed:users:writer
    global: true
    # <list> list of teams the role should be assigned to
    teams:
      - name: 'user editors'
        orgId: 1
```

Supported settings

The following sections detail the supported settings for roles and built-in role assignments.

- Refer to Permissions({{< relref "./permissions.md#action-definitions" >}}) for full list of valid permissions.

- Check [Custom roles]({{< relref "./roles.md#custom-roles" >}}) to understand attributes for roles.
- The [default org ID]({{< relref "../../administration/configuration#auto_assign_org_id" >}}) is used if `orgId` is not specified in any of the configuration blocks.

Validation rules

A basic set of validation rules are applied to the input `yaml` files.

Roles

- `name` must not be empty
- `name` must not have `fixed:` prefix.

Permissions

- `name` must not be empty

Built-in role assignments

- `name` must be one of the Organization roles (`Viewer`, `Editor`, `Admin`) or `Grafana Admin`.
- When `orgId` is not specified, it inherits the `orgId` from `role`. For global roles the default `orgId` is used.
- `orgId` in the `role` and in the assignment must be the same for none global roles.

Role deletion

- Either the role `name` or `uid` must be provided