OIDC Group Mappings

- Static Mappings
- Dynamic Mappings

Forgejo supports configuring group team mappings for OIDC authentication sources that map OIDC groups to Forgejo organizations and teams. These mappings allow you to automatically add a user in an OIDC group to one or multiple organization teams. For example, you can create a mapping in an authentication source that automatically adds a user in the OIDC group `Developers` to the teams `MyForgejoTeam1` and `MyForgejoTeam2` in organization `MyForgejoOrganization`.

There are two types of mappings: static mappings and dynamic mappings. Static mappings allow for a static configuration that maps a fixed set of OIDC groups each to a fixed set Forgejo organizations and teams. Dynamic mappings allow for a more advanced configuration that uses placeholders and simple pattern matching to map OIDC groups to organization teams. Both types of mappings can be used at the same time. You can find more information about both types below.

Static Mappings

You can configure static mappings for an authentication source at `https://<your-instance>/admin/auths` with the following fields:

- **Claim name providing group names for this source**: this defines the claim name for groups in the OIDC token, e.g., `groups`.
- **Map claimed groups to organization teams**: this contains a JSON object that defines the mappings, see example below.
- **Remove users from synchronized teams if user does not belong to corresponding group**: this defines whether a user is automatically removed from teams if there are OIDC group mappings (in the JSON object above) for the teams but the user is not a member of the respective OIDC group.

The following example shows the JSON object used to configure the mappings:

```
{
  "developer": {
    "MyForgejoOrganization": ["DeveloperTeam1", "DeveloperTeam2"]
```

```
  },
  "admin": {
    "MyForgejoOrganization": ["AdminTeam"],
    "OtherOrganization": ["admins"]
  }
}
```

In this example, a user in the OIDC group `developer` is automatically added to the teams `DeveloperTeam1` and `DeveloperTeam2` of organization `MyForgejoOrganization`. Additionally, a user in OIDC group `admin` is added to the team `AdminTeam` in the same organization `MyForgejoOrganization` and to team `admins` in organization `OtherOrganization`.

The example shows the format of the JSON object: it contains the name of each OIDC group (`developer`, `admin`). For each OIDC group it contains the names of the mapped organizations (`MyForgejoOrganization`, `OtherOrganization`) as well as the corresponding list of mapped teams in each organization (`DeveloperTeam1, DeveloperTeam2`, `AdminTeam`, `admins`).

Dynamic Mappings

You can configure dynamic mappings for an authentication source at `https://<your-instance>/admin/auths` with the following fields:

- **Claim name providing group names for this source**: this defines the claim name for groups in the OIDC token, e.g., `groups`.
- **Dynamically add users to teams based on dynamic group mappings**: this contains a JSON object that defines the mappings, see example below.
- **Dynamically remove users from all teams the user is not added to based on group mappings**: this defines whether a user is automatically removed from each team the user is currently a member of if the user is not added to the respective team by a static or dynamic group mapping. If enabled, OIDC users are only members of teams to which they are added by group mappings.

The following example shows the JSON object used to configure the mappings:

```
["group-{org}-{team}", "other:{org}/{team}"]
```

In this example, a user in OIDC group `group-MyForgejoOrganization-Developers` is added to team `Developers` in organization `MyForgejoOrganization`. A user in group `group-MyForgejoOrganization-admins` is added to team `admins` in the same organization `MyForgejoOrganization`. Additionally, a user in group `other:other-organization/other-team1` is added to team `other-team1` in organization `other-organization`.

The example shows the format of the JSON object: It is a list of patterns that are matched against OIDC group names. Each pattern contains the placeholders `{org}` and `{team}`. The placeholder `{org}` defines the position where the name of the organization is expected in the name of the OIDC group. `{team}` defines the position of the team name in the OIDC group. If a pattern matches an OIDC group name, the organization and team names are extracted from the OIDC group name at the respective positions. The placeholders must be separated from the rest of the pattern by characters that do not occur in the organization and team names (see note below).

Note: For this to work, you need to be careful when choosing OIDC group names and defining the patterns. The organization and team names should not use separators that are also used in other places of the patterns or otherwise the pattern matching will not work properly. For example, if you have group name `group-org-one-team-one` and pattern `group-{org}-{team}` it is not clear what the organization and team names are (`org`, `org-one`, `team-one`, `one-team-one`,...).

□□□ #2
□ □□□ □□□ 2026-10-31 13:33:28 UTC
□ □□□ □□□ 2026-10-31 13:33:29 UTC