Mopuan Roles and Permissions

Usage guide for the Mopuan integration

opsenseapi · Version 1.0 · 28 October 2026

1. What these services are for

In the OpSense portal every employee has one **role**. A role has:

- a **name** (for example "Supervisor" or "Housekeeper"),
- a **hierarchy level** that decides how much of the company's data the employee can see,
- a **permission matrix**: for each portal module, whether the role can view, create, edit and delete.

Employees are also linked to each other: an employee can have one or more **supervisors** (their manager) and a supervisor has a **team** (the employees who report to them).

Mopuan does not keep roles itself. These three services give you the OpSense roles, permissions and hierarchy so you can apply the same rules in Mopuan.

Service	Use it to
`SRVGetUserRoles`	Get one employee (or every employee) with their role, permissions, manager, team and visibility in a single object. Use it when an employee logs in or when you show one employee.
`SRVGetRoles`	Get the company's role list with each role's permissions. Use it to set up or refresh roles.
`SRVGetCompanyRoleData`	Get the whole company in one call: the role list plus every employee with their manager, team and visibility. Use it for a full sync, for example once a day.

All three are read-only. Calling them does not change anything.

2. Hierarchy levels

A **lower** level means **more** authority.

Level	Name	Can see
0	Super Admin	Every employee of the company
1	Admin	Every employee of the company
2	Supervisor	Themselves, their direct reports, and their direct reports' direct reports (two levels down)
3	User	Only themselves

Level 0 is used only by the built-in "Admin" role. Roles such as a CEO or a general manager, who must see everyone, are created in the portal at level 1.

An employee marked **only assigned** in the portal sees only themselves, whatever their role's level.

You do not need to work out visibility yourself. Every employee object has a ready `visibility` field (see section 6).

3. Authentication

Send the company's `api_key` as a normal request parameter, as a POST field, the same as for the other OpSense services. Do not send it as an HTTP header or in the URL.

The key decides which company the call is for. You only receive that company's roles and employees.

> **Call these services from your server only.** The `api_key` gives access to the whole company's employees and permissions. Never put it in the Mopuan mobile app or in a web page. If a mobile app needs an employee's permissions, it should ask your server, and your server calls `SRVGetUserRoles` for the employee who is logged in.

Situation	Response
`api_key` missing	`error_code` 10015
`api_key` invalid or inactive	`error_code` 10016

Every response is wrapped the same way:

```
{"result": { ... success or error body ... }}
```

4. Requests

```
POST https://<api-host>/v1/<service>
Content-Type: application/x-www-form-urlencoded
```

Send the parameters as **form fields**. A JSON request body is not read.

SRVGetUserRoles

Field	Required	Notes
`api_key`	**Required**	See Authentication.
`mopuan_user_id`	Optional	Return only this employee. Your Mopuan employee id.
`user_id`	Optional	Return only this employee, by OpSense user id. Ignored if `mopuan_user_id` is sent.

Without either filter you get every active employee.

```
curl -X POST "https://<api-host>/v1/SRVGetUserRoles" \
 -d "api_key=<api_key>" \
 -d "mopuan_user_id=345"
```

SRVGetRoles

Field	Required	Notes
`api_key`	**Required**	See Authentication.
`reference_id`	Optional	Return only this role, by role id.

```
curl -X POST "https://<api-host>/v1/SRVGetRoles" \
 -d "api_key=<api_key>"
```

SRVGetCompanyRoleData

Field	Required	Notes
`api_key`	**Required**	See Authentication.

```
curl -X POST "https://<api-host>/v1/SRVGetCompanyRoleData" \
 -d "api_key=<api_key>"
```

5. Role object

Returned in `roles` by `SRVGetRoles` and `SRVGetCompanyRoleData`, and inside each employee as `role` by `SRVGetUserRoles`.

```
{
 "id": 5,
 "name": "Supervisor",
 "hierarchy_level": 2,
 "hierarchy_name": "Supervisor",
 "is_global": false,
 "active": true,
 "user_count": 4,
 "permissions": {
   "work-orders": {"view": true, "create": true, "edit": true, "delete": false},
   "users-list":  {"view": false, "create": false, "edit": false, "delete": false}
 }
}
```

Field	Description
`id`	Role id. Employees point to it with `role_id`.
`name`	Role name as set in the portal.
`hierarchy_level`	0 to 3, see section 2.

`hierarchy_name`	Name of the level: Super Admin, Admin, Supervisor or User.
`is_global`	`true` only for the built-in "Admin" role, which every company has.
`active`	`false` if the role has been switched off in the portal.
`user_count`	Number of active employees with this role. Not returned by `SRVGetUserRoles`.
`permissions`	One entry for every module listed in `sections`, keyed by the module key. Each has `view`, `create`, `edit` and `delete` as `true` or `false`. A module the role has no access to is still listed, with all four set to `false`.

`SRVGetRoles` and `SRVGetCompanyRoleData` also return:

Field	Description
`hierarchy_levels`	The table in section 2: `level`, `name` and `scope` (`all`, `team` or `self`).
`sections`	Every portal module: `key` (used in `permissions`) and `name`.

Use the module `key`, not its name. The key does not change; the name may.

6. Employee object

Returned in `users` by `SRVGetCompanyRoleData` and in `data` by `SRVGetUserRoles`.

```
{
 "user_id": 12,
 "mopuan_user_id": "345",
 "name": "Jane Doe",
 "role_id": 5,
 "only_assigned": false,
 "supervisors": [{"user_id": 3, "mopuan_user_id": "101"}],
 "direct_reports": [
   {"user_id": 20, "mopuan_user_id": "410"},
   {"user_id": 21, "mopuan_user_id": null}
 ],
 "visibility": {
   "scope": "team",
   "user_ids": [12, 20, 21, 30],
   "mopuan_user_ids": ["345", "410", "512"]
 }
}
```

Field	Description
`user_id`	OpSense user id.
`mopuan_user_id`	Your employee id, or `null` if this employee is not linked to Mopuan yet. Such employees are still listed so the hierarchy has no gaps.

`name`	First name and surname.
`role_id`	The employee's role (`SRVGetCompanyRoleData` only).
`role`	The full role object from section 5 (`SRVGetUserRoles` only, instead of `role_id`).
`only_assigned`	`true` if the portal limits this employee to their own data.
`supervisors`	Who this employee reports to. Can be empty or have more than one entry. A supervisor who has their own manager has their own `supervisors` list.
`direct_reports`	Employees who report directly to this employee. Can be empty.
`visibility.scope`	`all`: sees every employee of the company. `team`: sees the employees in `user_ids`. `self`: sees only themselves.
`visibility.user_ids`	The OpSense ids this employee can see, including themselves. Not sent when `scope` is `all`.
`visibility.mopuan_user_ids`	The same list as your employee ids. Employees not linked to Mopuan are left out. Not sent when `scope` is `all`.

Only active employees are returned. Supervisor and team links to inactive or deleted employees are left out.

The visibility rules are the same ones the portal uses, checked in this order:

1. `only_assigned` is `true`: `self`.
2. Level 0 or 1: `all`.
3. Level 2: `team`. The employee, their direct reports, and those reports' direct reports. Further levels down are not included.
4. Level 3: `self`. This is also true for a level 3 employee who has someone reporting to them.

7. Errors

error_code	Meaning	Retry?
10015	`api_key` missing.	No. Fix the request.
10016	`api_key` invalid or inactive.	No. Check the key.
10020	`SRVGetRoles`: `reference_id` is not a role of this company. `SRVGetUserRoles`: the employee is not an active employee of this company.	No.
10099	The request could not be completed on our side.	Yes

```
{"result": {"error_code": "10020", "error_text": "User not found."}}
```

8. When to call

- **Employee logs in to Mopuan:** call `SRVGetUserRoles` with their `mopuan_user_id`. One call gives their permissions and whose data they can see.
- **Full sync:** call `SRVGetCompanyRoleData` once a day, or after you sync employees. Replace what you stored before with the new response.
- **Role setup screens:** call `SRVGetRoles`.

Roles, permissions and supervisors are changed in the OpSense portal. These services do not tell you when that happens, so do not keep a copy for longer than one day.