Setting up OIDC SSO Single Sign-on integration

Overview

PatchMon supports OpenID Connect (OIDC) authentication, allowing users to log in via an external Identity Provider (IdP) instead of, or in addition to, local username/password credentials.

Supported Providers

Any OIDC-compliant provider works, including:

- Authentik
- Keycloak
- Okta
- Azure AD (Entra ID)
- Google Workspace
- And others

What You Get

- **SSO login** via a configurable button on the login page
- **Automatic user provisioning** on first login (no need to create accounts manually)
- **Group-based role mapping** so your IdP controls who is an admin, user, or readonly viewer
- **Optional** - disable local password login entirely and enforce SSO for all users

---

Prerequisites

- PatchMon already installed and running
- An OIDC-compatible Identity Provider with an OAuth2/OIDC application configured
- HTTPS in production (OIDC routes enforce HTTPS when `NODE_ENV=production`)

Step 1 - Create an OIDC Application in Your IdP

Create a new OAuth2 / OIDC application in your Identity Provider with the following settings:

Setting	Value
Application type	Web application / Confidential client
Redirect URI	`https://patchmon.example.com/api/v1/auth/oidc/callback`
Scopes	`openid`, `email`, `profile`, `groups`
Grant type	Authorization Code
Token endpoint auth	Client Secret (Basic)

After creating the application, note the **Client ID** and **Client Secret** as you'll need both.

> **Tip:** If you plan to use group-based role mapping, ensure your IdP includes the `groups` claim in the ID token. In Authentik, this is enabled by default. In Keycloak, you may need to add a "Group Membership" mapper to the client scope.

Provider-Specific Notes

Authentik:

- Create an OAuth2/OIDC Provider, then create an Application linked to it
- Issuer URL format: `https://auth.example.com/application/o/patchmon/`
- Groups are included via the `groups` or `ak_groups` claim (both are supported)

Keycloak:

- Create a Client with Access Type `confidential`
- Issuer URL format: `https://keycloak.example.com/realms/your-realm`
- Add a "Group Membership" protocol mapper to include groups in the token

Okta / Azure AD:

- Create an OIDC Web Application
- Ensure groups are included in the ID token claims

Step 2 - Configure PatchMon

Add the following environment variables to your `.env` file (for Docker deployments) or your backend environment.

Required Variables

```
OIDC_ENABLED=true
OIDC_ISSUER_URL=https://auth.example.com/application/o/patchmon/
OIDC_CLIENT_ID=your-client-id
OIDC_CLIENT_SECRET=your-client-secret
OIDC_REDIRECT_URI=https://patchmon.example.com/api/v1/auth/oidc/callback
```

Variable	Description
`OIDC_ENABLED`	Set to `true` to enable OIDC
`OIDC_ISSUER_URL`	Your IdP's issuer / discovery URL
`OIDC_CLIENT_ID`	Client ID from your IdP application
`OIDC_CLIENT_SECRET`	Client secret from your IdP application
`OIDC_REDIRECT_URI`	Must match exactly what you configured in your IdP

Optional Variables

```
OIDC_SCOPES=openid email profile groups
OIDC_AUTO_CREATE_USERS=true
OIDC_DEFAULT_ROLE=user
OIDC_DISABLE_LOCAL_AUTH=false
OIDC_BUTTON_TEXT=Login with SSO
```

Variable	Default	Description
`OIDC_SCOPES`	`openid email profile groups`	Space-separated scopes to request. Include `groups` for role mapping
`OIDC_AUTO_CREATE_USERS`	`true`	Automatically create a PatchMon account on first OIDC login
`OIDC_DEFAULT_ROLE`	`user`	Role assigned when a user doesn't match any group mapping
`OIDC_DISABLE_LOCAL_AUTH`	`false`	When `true`, hides the username/password fields and only shows the SSO button

Variable	Default	Description
`OIDC_BUTTON_TEXT`	`Login with SSO`	Label shown on the SSO login button

Step 3 - Group-Based Role Mapping (Optional)

Map your IdP groups to PatchMon roles so that role assignments stay in sync with your directory. Group matching is **case-insensitive**.

Role Hierarchy

PatchMon checks group membership in this order (highest priority first):

PatchMon Role	Required IdP Group(s)	Description
Super Admin	Member of BOTH `OIDC_ADMIN_GROUP` AND `OIDC_SUPERADMIN_GROUP`	Full access including system settings
Admin	Member of `OIDC_ADMIN_GROUP`	Full access
Host Manager	Member of `OIDC_HOST_MANAGER_GROUP`	Manage hosts and groups
User	Member of `OIDC_USER_GROUP`	Standard access with data export
Readonly	Member of `OIDC_READONLY_GROUP`	View-only access
Default	None of the above	Gets `OIDC_DEFAULT_ROLE` (defaults to `user`)

Environment Variables

```
OIDC_ADMIN_GROUP=PatchMon Admins
OIDC_USER_GROUP=PatchMon Users
OIDC_SUPERADMIN_GROUP=PatchMon SuperAdmins
OIDC_HOST_MANAGER_GROUP=PatchMon Host Managers
OIDC_READONLY_GROUP=PatchMon Readonly
OIDC_SYNC_ROLES=true
```

Variable	Description
`OIDC_ADMIN_GROUP`	IdP group name that maps to Admin role
`OIDC_USER_GROUP`	IdP group name that maps to User role

Variable	Description
`OIDC_SUPERADMIN_GROUP`	IdP group name that maps to Super Admin (requires **both** this and Admin group)
`OIDC_HOST_MANAGER_GROUP`	IdP group name that maps to Host Manager role
`OIDC_READONLY_GROUP`	IdP group name that maps to Readonly role
`OIDC_SYNC_ROLES`	When `true` (default), the user's role is updated on every login based on current group membership. When `false`, the role is only set on first login

You only need to define the groups you intend to use. Any variables left unset are simply ignored.

Step 4 - Restart PatchMon

After updating your `.env` file, restart the backend so it discovers your OIDC provider on startup:

```
# Docker
docker compose restart backend

# Or if rebuilding
docker compose up -d --force-recreate backend
```

Check the backend logs to confirm OIDC initialised:

```
docker compose logs backend | grep -i oidc
```

You should see:

```
Discovering OIDC configuration from: https://auth.example.com/...
OIDC Issuer discovered: https://auth.example.com/...
OIDC client initialized successfully
```

If you see `OIDC is enabled but missing required configuration`, double-check your environment variables.

Step 5 - Test the Login

1. Open PatchMon in your browser
2. You should see a **"Login with SSO"** button (or your custom `OIDC_BUTTON_TEXT`)

3. Click it and you'll be redirected to your IdP
4. Authenticate with your IdP credentials
5. You'll be redirected back to PatchMon and logged in

If `OIDC_AUTO_CREATE_USERS` is `true`, a PatchMon account is created automatically using your email address. The username is derived from the email prefix (e.g. `john.doe@example.com` becomes `john.doe`).

First-Time Setup (No Users Exist Yet)

When PatchMon has no users in the database, it displays a setup wizard. If you're using OIDC-only mode (`OIDC_DISABLE_LOCAL_AUTH=true`), you have two options:

Option A - Log In via OIDC (Recommended)

1. Ensure your IdP user is in the admin group (e.g. `PatchMon Admins`)
2. Set `OIDC_AUTO_CREATE_USERS=true`
3. Click the SSO button and the first user will be created with the role determined by your group mapping

Option B - Disable OIDC for the first Admin

If the setup wizard blocks access then you can create a local Admin on first setup then enable/setup OIDC after that. You can remove the first admin user but you should be Super Admin Role.

What Syncs from Your IdP

On every OIDC login, PatchMon automatically syncs the following from your Identity Provider:

- **Role** (if `OIDC_SYNC_ROLES=true`) - based on group membership
- **Avatar / profile picture** - if the `picture` claim is present
- **First name and last name** - from `given_name` and `family_name` claims
- **Email** - used for matching and account linking

Account Linking

If a local PatchMon user already exists with the same email as the OIDC user, PatchMon will automatically link the accounts, but only if the email is marked as **verified** by the IdP. This

prevents account takeover via unverified emails.

Disabling Local Authentication

To enforce SSO for all users, set:

```
OIDC_DISABLE_LOCAL_AUTH=true
```

This hides the username/password fields on the login page and only shows the SSO button. Local authentication is only actually disabled if OIDC is also enabled and successfully initialised. This safety check prevents you from being locked out if OIDC is misconfigured.

> **Important:** Ensure at least one OIDC user has admin access before enabling this, or you may lose the ability to manage PatchMon.

Complete Example Configuration

Authentik

```
# .env
OIDC_ENABLED=true
OIDC_ISSUER_URL=https://authentik.example.com/application/o/patchmon/
OIDC_CLIENT_ID=patchmon
OIDC_CLIENT_SECRET=your-client-secret-here
OIDC_REDIRECT_URI=https://patchmon.example.com/api/v1/auth/oidc/callback
OIDC_SCOPES=openid email profile groups
OIDC_AUTO_CREATE_USERS=true
OIDC_DEFAULT_ROLE=user
OIDC_BUTTON_TEXT=Login with Authentik
OIDC_ADMIN_GROUP=PatchMon Admins
OIDC_USER_GROUP=PatchMon Users
OIDC_SYNC_ROLES=true
```

Keycloak

```
# .env
OIDC_ENABLED=true
OIDC_ISSUER_URL=https://keycloak.example.com/realms/your-realm
OIDC_CLIENT_ID=patchmon
OIDC_CLIENT_SECRET=your-client-secret-here
OIDC_REDIRECT_URI=https://patchmon.example.com/api/v1/auth/oidc/callback
OIDC_SCOPES=openid email profile groups
OIDC_AUTO_CREATE_USERS=true
OIDC_DEFAULT_ROLE=user
OIDC_BUTTON_TEXT=Login with Keycloak
OIDC_ADMIN_GROUP=PatchMon Admins
OIDC_USER_GROUP=PatchMon Users
OIDC_SYNC_ROLES=true
```

Troubleshooting

OIDC Not Initialising

Logs show: `OIDC is enabled but missing required configuration`

All four required variables must be set: `OIDC_ISSUER_URL`, `OIDC_CLIENT_ID`, `OIDC_CLIENT_SECRET`, `OIDC_REDIRECT_URI`. Check for typos or empty values.

SSO Button Not Appearing

The button only appears if OIDC is both enabled (`OIDC_ENABLED=true`) **and** successfully initialised. Check backend logs for OIDC errors. Common causes:

- PatchMon cannot reach the IdP (DNS / firewall issue)
- Issuer URL is incorrect
- IdP's `.well-known/openid-configuration` endpoint is not accessible

"Authentication Failed" After Redirect

- Verify the **Redirect URI** in your IdP matches `OIDC_REDIRECT_URI` exactly (including trailing slashes)
- Ensure cookies are not being blocked (OIDC uses httpOnly cookies for session state)
- Check that your IdP supports PKCE (PatchMon uses S256 code challenge)

"Session Expired" Error

The OIDC session has a 10-minute window between initiating login and completing the callback. If the user takes too long at the IdP, the session expires. Simply try logging in again.

User Gets Wrong Role

- Check that the `groups` scope is included in `OIDC_SCOPES`
- Verify your IdP is including groups in the ID token (not just the access token)
- Check backend logs as they show which groups were received: `OIDC groups found: [...]`
- If logs show `No groups found in OIDC token`, configure your IdP to include the groups claim
- Group matching is case-insensitive, so `patchmon admins` matches `PatchMon Admins`

"User Not Found" Error

`OIDC_AUTO_CREATE_USERS` is set to `false` and no matching PatchMon account exists. Either enable auto-creation or create the user account manually in PatchMon first (the email must match).

Debug Logging

For detailed OIDC troubleshooting, enable debug logging:

```
LOG_LEVEL=debug
```

Then check the backend logs:

```
docker compose logs -f backend | grep -i oidc
```

Security Notes

- **HTTPS is enforced** for OIDC login and callback routes when `NODE_ENV=production`
- **PKCE (S256)** is used for all authorization code exchanges
- **Tokens are stored in httpOnly cookies**, not localStorage, to prevent XSS attacks
- **Client secrets** should never be committed to version control
- **Account linking** only occurs when the IdP reports the email as verified
- **Role sync** can be disabled (`OIDC_SYNC_ROLES=false`) if you prefer to manage roles manually in PatchMon after first login

---

Revision #1
Created 2026-02-12 02:42:13 UTC by Iby
Updated 2026-02-12 02:49:44 UTC by Iby