OIDC Authentication

With release 5.3.0, TeamCal Neo supports Single Sign-On (SSO) login using **OpenID Connect (OIDC)**. This allows users to authenticate through an external Identity Provider (IdP) such as Keycloak, Microsoft Entra (Azure AD), Okta, or any other OIDC-certified service — without entering a TeamCal Neo password.

How it works

TeamCal Neo uses the **Authorization Code Flow**. When a user clicks "Login with SSO":

1. TeamCal Neo redirects the browser to the IdP's login page.
2. The user authenticates at the IdP (the password is never sent to TeamCal Neo).
3. The IdP redirects back to TeamCal Neo with a short-lived code.
4. TeamCal Neo exchanges the code for an **ID Token** (server-to-server).
5. The ID Token contains claims — in particular `sub` (a permanent unique ID) and `preferred_username` (the human-readable login name).
6. TeamCal Neo looks up the local user account by `sub`, then by `preferred_username` as a fallback.
7. On first match via `preferred_username`, the `sub` is stored for future lookups so IdP username renames do not break the link.
8. If no matching local account is found, login is rejected.
9. On success, a standard session cookie is set — the rest of the application is unchanged.

Prerequisites

- A running OIDC-compatible Identity Provider (Keycloak, Entra, Okta, Google, etc.)
- A **Client** registered on the IdP for TeamCal Neo (see below)
- PHP Composer package `jumbojett/openid-connect-php` (included in the 5.3.0 distribution)

Configuration

OIDC is configured via the `.env` file. Add the following entries (see `.env.example` for reference):

```
OIDC_YES=1
OIDC_PROVIDER_URL=https://keycloak.example.com/realms/teamcalneo
OIDC_CLIENT_ID=teamcal-neo
OIDC_CLIENT_SECRET=your-client-secret-here
OIDC_REDIRECT_URI=https://your-teamcal-server.com/index.php?action=oidccallback
```

Variable	Description
`OIDC_YES`	`1` to enable OIDC, `0` to disable
`OIDC_PROVIDER_URL`	Base URL of the IdP realm. The library fetches `/.well-known/openid-configuration` from it automatically.
`OIDC_CLIENT_ID`	The client ID registered at the IdP
`OIDC_CLIENT_SECRET`	The client secret from the IdP
`OIDC_REDIRECT_URI`	The callback URL back to TeamCal Neo — must match the redirect URI configured at the IdP

When `OIDC_YES=1`:

- **LDAP authentication is automatically disabled**, even if `LDAP_YES=1` is also set.
- The login page shows a "Login with SSO" button and hides the standard username/password form for regular users.
- The `admin` account always authenticates locally, regardless of the OIDC setting.
- The application's built-in two-factor authentication (TOTP) is bypassed for OIDC-authenticated users. MFA is the responsibility of the IdP.

User management

OIDC authentication does **not** auto-provision new user accounts. A local TeamCal Neo account must exist before a user can log in via OIDC. This is by design — local accounts are where group memberships, roles, and absence permissions are managed.

> ⚠ **Username matching is critical.** When OIDC is active, the **Username** field of a local TeamCal Neo account must exactly match the `preferred_username` claim the IdP sends in the ID Token. If the names do not match, the user will be rejected with "No local account found" even though the IdP authentication succeeded.

The user add and user edit screens display a reminder when OIDC is active.

After the first successful OIDC login, TeamCal Neo stores the IdP's permanent `sub` identifier against the local account. Subsequent logins look up the account by `sub` first, so IdP username renames do not orphan the account.

Behaviour overview

Scenario	Result
OIDC active, known user clicks "Login with SSO"	Login succeeds; session cookie set
OIDC active, unknown user (no local account)	Rejected: "No local account found"
OIDC active, user tries to log in via the local form	Rejected: "Local login is disabled while SSO is active"
OIDC active, `admin` logs in via the local form	Allowed — `admin` is always local-only
OIDC active, locked or unverified account authenticates at IdP	Rejected with the corresponding account status message
OIDC active, user logs out	Local session cookie cleared; IdP session remains active (no Single Logout)
OIDC active, `LDAP_YES=1` also set in `.env`	LDAP is ignored; OIDC takes precedence

---

Keycloak setup for testing

Keycloak (https://www.keycloak.org) is a free, open-source IdP that runs in Docker. It is the recommended option for testing OIDC before connecting to a production IdP such as Entra or Okta.

Step 1 — Start Keycloak in Docker

```
docker run -d \
  --name keycloak \
  -p 8180:8080 \
  -e KEYCLOAK_ADMIN=admin \
  -e KEYCLOAK_ADMIN_PASSWORD=admin \
  quay.io/keycloak/keycloak:latest \
  start-dev
```

After about a minute, the Keycloak admin console is available at `http://<host>:8180`.

Step 2 — Create a Realm

1. Log in to the Keycloak admin console.
2. Open the realm dropdown (top-left) → **Create Realm**.
3. Name it `teamcalneo` → **Create**.

All following steps are done inside the `teamcalneo` realm.

Step 3 — Create the TeamCal Neo client

1. Go to **Clients** → **Create client**.
2. **Client ID**: `teamcal-neo`, **Client type**: `OpenID Connect` → **Next**.
3. Enable **Client authentication** → **Next**.
4. **Valid redirect URIs**: `http://<teamcal-server>/index.php?action=oidccallback` → **Save**.
5. Open the **Credentials** tab and copy the **Client secret**.

Step 4 — Create a test user

1. Go to **Users** → **Create new user**.
2. Set **Username** to a value that exactly matches a username in the TeamCal Neo database.
3. Open the **Credentials** tab → **Set password** (disable "Temporary").

Step 5 — Verify the discovery document

Open the following URL in a browser to confirm Keycloak is reachable and OIDC-ready:

```
http://<keycloak-host>:8180/realms/teamcalneo/.well-known/openid-configuration
```

If a JSON document is returned, Keycloak is working correctly.

Step 6 — Configure TeamCal Neo

```
OIDC_YES=1
OIDC_PROVIDER_URL=http://<keycloak-host>:8180/realms/teamcalneo
OIDC_CLIENT_ID=teamcal-neo
OIDC_CLIENT_SECRET=<secret from Step 3>
OIDC_REDIRECT_URI=http://<teamcal-server>/index.php?action=oidccallback
```

Connecting to production Identity Providers

Once the feature works against Keycloak, switching to a production IdP only requires updating the `.env` file. No code changes are needed.

Identity Provider	`OIDC_PROVIDER_URL` value
Microsoft Entra (Azure AD)	`https://login.microsoftonline.com/<tenant-id>/v2.0`
Okta	`https://<your-domain>.okta.com`
Google	`https://accounts.google.com`
Keycloak	`https://<host>/realms/<realm-name>`

The OIDC library fetches the discovery document from `<OIDC_PROVIDER_URL>/.well-known/openid-configuration` at runtime and adapts to the IdP automatically.

Troubleshooting

Symptom	Likely cause
"No local account found" after IdP login	The `preferred_username` in the IdP does not match any TeamCal Neo username
"OIDC: Provider connection error" on the login page	`OIDC_PROVIDER_URL` is unreachable or incorrect
Redirect loop or state mismatch error	The session cookie `SameSite` attribute must be `Lax` (not `Strict`) — already set correctly in 5.3.0
LDAP login still works when OIDC is enabled	Check that `OIDC_YES=1` is set and not overridden by a cached `.env`
User is rejected despite a matching username	Check whether the account is locked or unverified in TeamCal Neo user management

For detailed login event logs, check the TeamCal Neo **System Log** (`Administration → System Log`). Keycloak also records all authentication events under **Realm settings** → **Events**.

Last updated: 2026-06-12 15:38