

OPENID Keycloak Anbindung Proxmox

Du möchtest dich gerne für unsere Hilfe erkenntlich zeigen 🙏 . Gerne. Wir bedanken uns bei dir für deine Spende! ☐

[Spenden](#)

Zum frei verfügbaren [Apt-Repository](#)



GITLAB:

Die folgende Dokumentation zeigt die Keycloakanbindung von Proxmox inkl. Login berechtigten Gruppen. Als Backend wird [LDAP von UCS \(Univention\)](#) verwendet. Für das ganze Vorhaben wird eine ähnlich funktionierende Umgebung voraus gesetzt.

Verwendete Systeme/Software:

- Proxmox 8.1.4
- UCS 5.0-6 errata993
- Keycloak installiert am Primary Directory Node (ohne verteilter Datenbank) 23.0.7

Proxmoxclusternodes:

- **pve01.tux.lan**
- **pve02.tux.lan**
- **pve03.tux.lan**

Primary Directory Node: **dc1.tux.lan**

OpenID Client in Keycloak hinzufügen

Unter dem Realm „ucs“ wird ein neuer Client names „proxmox-cluster01“ hinzugefügt. Die Basiseinrichtung erfolgt in 3 Schritten:

1. Erstellen eines neuen Clients

The screenshot shows the Keycloak Admin Console interface. On the left, a sidebar menu is visible with the 'ucs' realm selected. The main content area is titled 'Clients' and contains a search bar, a 'Create client' button (highlighted with a red arrow), and an 'Import client' button. Below this, a table lists existing clients:

Client-ID	Name
account	\$(client_account)
account-console	\$(client_account-console)

2. Setzen des „Client type“ und der „Client-ID“

Clients > Create client

Create client

Clients are applications and services that can request authentication of a user.

- General settings
- Capability config
- Login settings

Client type

Client-ID

Name

Beschreibung

Always display in UI Off

Clients > Create client

Create client

Clients are applications and services that can request authentication of a user.

- General settings
- Capability config
- Login settings

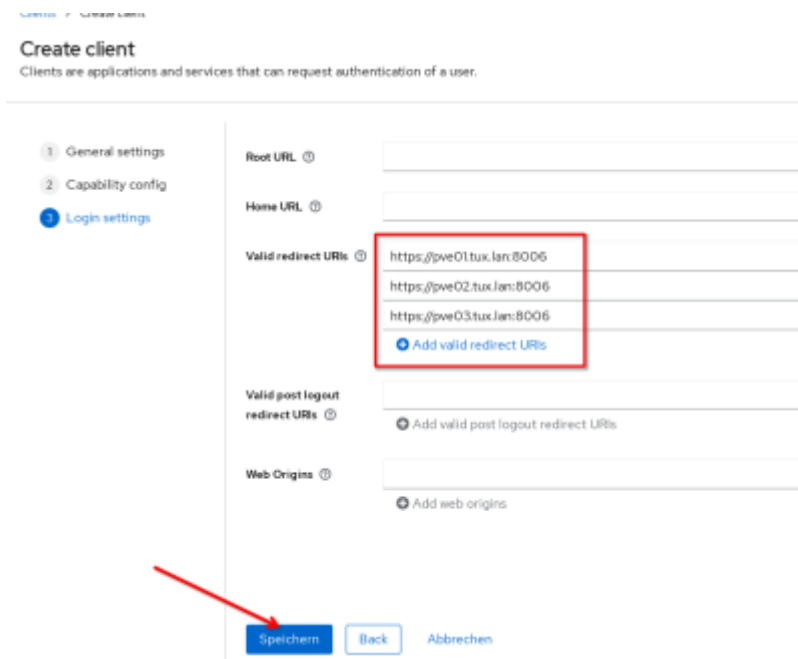
Client authentication Off

Authorization Off

Authentication flow

- Standard flow
- Implicit flow
- OAuth 2.0 Device Authorization Grant
- OIDC CIBA Grant
- Direct access grants
- Service accounts roles

3. Setzen der „Valid redirect URI's“



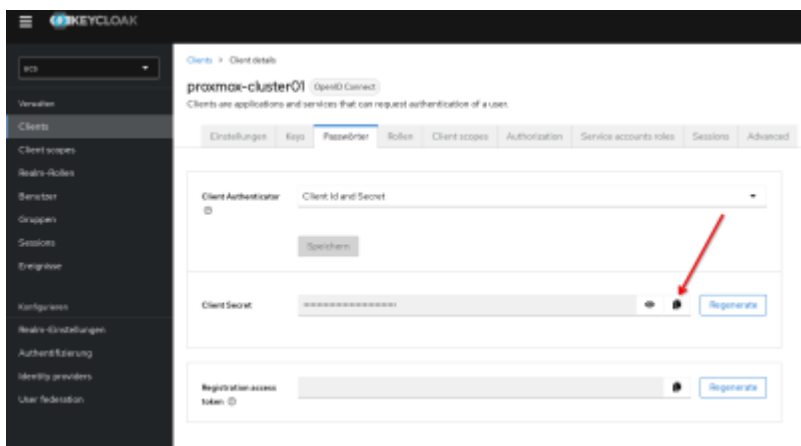
Für unser späteres Vorhaben „nur bestimmte Gruppen zu zulassen“, müssen nach dem „Speichern“ noch zwei weitere Optionen unter „**Einstellungen**“ aktiviert werden. Dies schaltet weitere Funktionen frei.



Damit wäre die Basiseinrichtung abgeschlossen.

Konfiguration OpenID auf Proxmox

Hier bedient man sich am besten der CMD. Bevor man dies tut muss man sich aber noch das „**Client Secret**“ kopieren.



Danach wird folgender Befehl auf der Rootshell von Proxmox abgesetzt:

```
pveum realm add tux.lan-SSO --type openid --issuer-url  
https://ucs-sso-ng.tux.lan/realms/ucs --client-id proxmox-cluster01 --  
client-key XXXXX --username-claim username
```

–autocreate wäre optional. Damit werden Benutzer beim Ersten Login automatisch angelegt. Ab dem Zeitpunkt ist der Login mittels SSO/SAML möglich. Man hat aber noch keine Rechte. [Berechtigungen müssen manuell im Proxmox Webinterface für den/die Benutzer hinzugefügt werden.](#) Berechtigungen werden „on the fly“ übernommen.

Die Empfehlung ist hier eine Gruppe im Proxmox Webinterface zu erstellen und den Benutzer dort einfach hinzuzufügen.

Datacenter → Permissions → Groups Gruppe anlegen, z.B. „admin“

Datacenter → Permissions → Users Gewünschten Benutzer bearbeiten und die Gruppe „admin“ zuweisen.

Einschränkung auf Gruppen

Um überhaupt zu den LDAP-Gruppen zu kommen, muss ein „**group-ldap-mapper**“ hinzugefügt werden. Hierzu wechselt man im Menü von Keycloak auf „**User federation**“ und bearbeitet den „**ldap-provider**“. Im TAB Mappers, fügt man nun den „**group-ldap-mapper**“ hinzu.

User federation > Einstellungen

LDAP

Einstellungen Mappers

Search for mapper → Add mapper

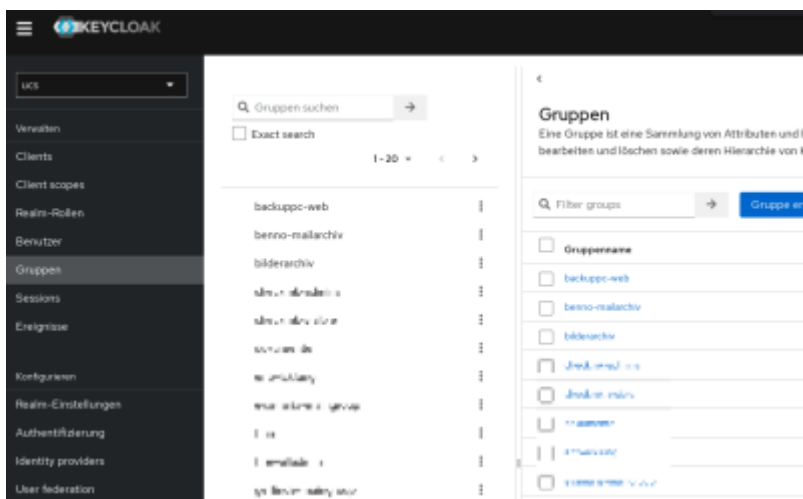
Name	Typ
creation date	user-attribute-ldap-mapper
displayName	user-attribute-ldap-mapper
email	user-attribute-ldap-mapper
entryUUID	user-attribute-ldap-mapper
first name	user-attribute-ldap-mapper
group-ldap-mapper	group-ldap-mapper
last name	user-attribute-ldap-mapper
modify date	user-attribute-ldap-mapper
second-mail	user-attribute-ldap-mapper
uid	user-attribute-ldap-mapper
Univention ldap mapper	univention-ldap-mapper
username	user-attribute-ldap-mapper

Der Inhalt wurde auf einen Default UCS-LDAP angepasst.

Attributbeschreibung	Attributname	Info
ID	auto generiert	
Name	group-ldap-mapper	
Mapper type	group-ldap-mapper	
LDAP Groups DN	cn=tux-groups,cn=groups,dc=tux,dc=lan	Beispiel
Group Name LDAP Attribute	cn	
Group Object Classes	posixGroup	
Preserve Group Inheritance	OFF	
Ignore Missing Groups	OFF	
Membership LDAP Attribute	memberUid	
Membership Attribute Type	UID	
Membership User LDAP Attribute	uid	
LDAP Filter	(&(uid=%s)(memberof=cn=proximoxi,cn=tux-groups,cn=groups,dc=tux,dc=lan))	Kann verwendet werden um noch granularer zu werden.
Mode	READ_ONLY	
User Groups Retrieve Strategy	LOAD_GROUPS_BY_MEMBER_ATTRIBUTE	
Member-Of LDAP Attribute	memberOf	

Attributbeschreibung	Attributname	Info
Mapped Group Attributes		
Drop non-existing groups during sync	OFF	
Groups Path	/	Dies zu Ändern macht bei vielen Gruppen vielleicht Sinn.

Danach **„Speichern“**, nochmal einsteigen und rechts oben auf **„Aktion → Sync LDAP groups to Keycloak“** anklicken. Damit sollte eine grüne Infomeldung aufpoppen wo die gesyncten Gruppen angezeigt werden. Damit sind unter **„Gruppen“** nun auch alle Gruppen und Groupmembers in Keycloak ersichtlich.



Userimport und automatischer Sync in Echtzeit (optional)

Dieser Schritt muss nicht durchgeführt werden. Keycloak schaut auch jedes mal gerne am LDAP Live nach welche Benutzer es gibt. Aus Performancegründe macht es bei größeren Installationen Sinn die Benutzer direkt in die lokal MariaDB zu syncen. Hier zu bearbeitet man wieder den **„ldap-provider“** und aktiviert bei **„Synchronization settings“** das Flag bei **Import users**. Danach einmal abspeichern.

Synchronization settings

Import users On

Sync Registrations Off

Batch size

Periodic full sync Off

Periodic changed users sync Off

Jetzt hat man in der rechten oberen Ecke unter „**Aktion**“ eine neue freigehaltete Funktion: „**Sync all users**“. Hier sollte wieder eine grüne Infomeldung aufpoppen wo die gesyncnten Benutzer angezeigt werden. Nach dieser Aktion möchte man auch noch den Livesync der User und Gruppenmitgliedschaften aktivieren. Hierzu noch das folgende Flag auf „On“ schalten.

Synchronization settings

Import users On

Sync Registrations Off

Batch size

Periodic full sync Off

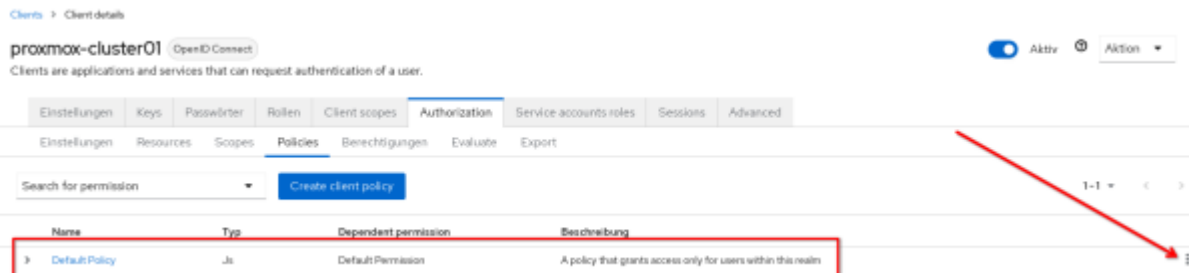
Periodic changed users sync On

Changed users sync period

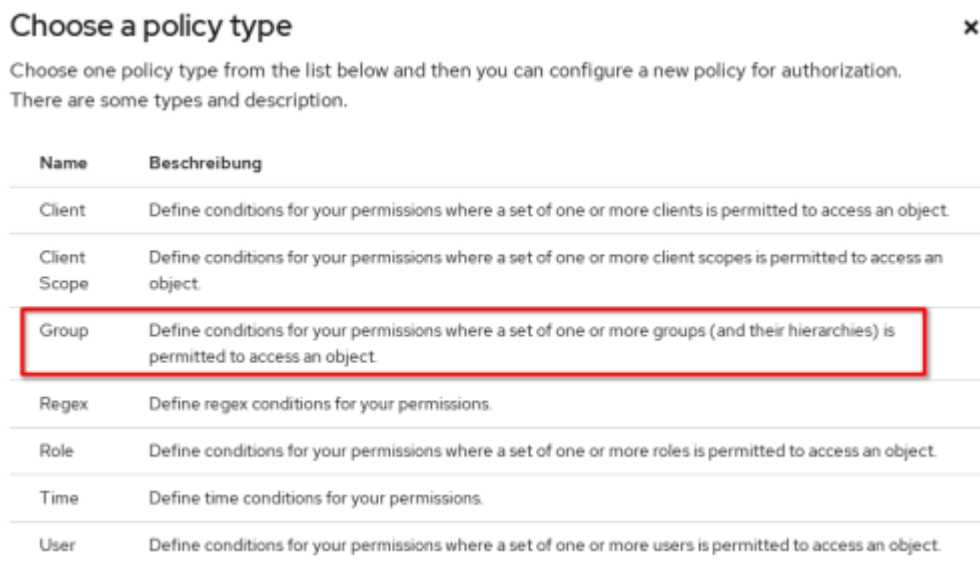
Beim Speichern der Einstellungen deaktiviert sich dieses Flag wieder. ...scheint aber zu funktionieren, weil die richtigen Info's in Keycloak angezeigt werden.

Einrichtung der Authorization im proxmox-cluster01 Client

Als erstes muss im Client die „Default Policy gelöscht werden“.



Nun fügen wir eine neue Group-Policy hinzu.



Create group policy

Name [?]


Beschreibung [?]

Groups claim [?]

Gruppen [?]

Gruppen	Extend to children
/testgruppebla	<input type="checkbox"/>
/gitlab	<input type="checkbox"/>

Logic [?] Positive Negative

 Die folgenden zwei Gruppen „testgruppebla“ und „gitlab“ sollen also berechtigt sein sich auf dem Cluster einloggen zu können.

Im nächsten Schritt fügen wir nun noch die Berechtigungen hinzu.


Clients > Client details

proxmox-cluster01 OpenID Connect

Clients are applications and services that can request authentication of a user.

Einstellungen Keys Passwörter Rollen Client scopes **Authorization** Service accounts roles Sessions Advanced

Einstellungen Resources Scopes Policies **Berechtigungen** Evaluate Export


No permissions
If you want to create a permission, please click the button below to create a resource-based or scope-based permission.

Clients > Client details > Create permission

Create resource-based permission

Name *

Beschreibung

Apply to resource type Off

Resources *

Policies

Decision strategy Affirmative Unanimous Consensus

Benutzer Evaluierung

Ein ganz bequemes Werkzeug ist die Benutzer Evaluierung. Dies befindet sich auch in der Clientkonfiguration direkt neben Berechtigungen. Damit ist es möglich Benutzerrechte live zu testen. Da Keycloak einen Cache betreibt, ist das Werkzeug nicht mehr weg zu denken.

Einstellungen Keys Passwörter Rollen Client scopes Authorization Service accounts roles Sessions Advanced

Einstellungen Resources Scopes Policies Berechtigungen Evaluieren Export

Identity information

Client

Benutzer *

Rollen

Identity information

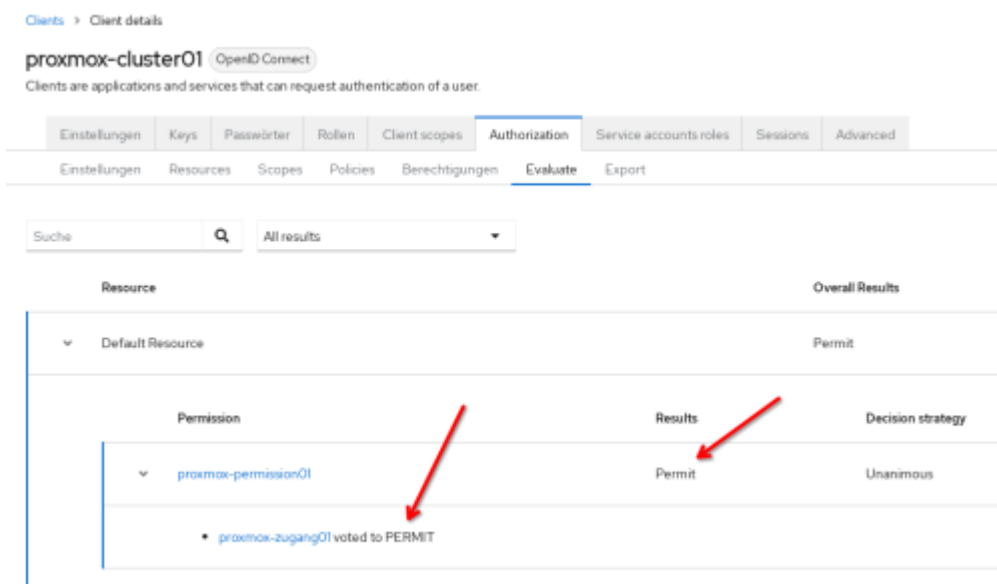
Apply to Resource Type Off

Resources and Scopes

Schlüssel	Wert
<input type="text" value="Select or type a key"/>	<input type="text" value="Select or type a key"/>

> Contextual Information

Hier kann man sehr gut erkennen das der Benutzer „harald“ nicht darf. Sehr gut. Wie sieht es nun aus wenn wir den Benutzer Harald in die Gruppe „gitlab“ werfen?



Und schon darf er sich einloggen.

Wie man sieht ist mit Keycloak/UCS/Proxmox schon einiges möglich. Und damit wäre die Konfiguration auch schon abgeschlossen.

Automatisch delegierte Berechtigungen über eine LDAP-Gruppe



Damit müssen keine Berechtigungen mehr dem Benutzern in Proxmox manuell zugewiesen werden.

 **Fix Me!**

<https://lists.proxmox.com/pipermail/pve-devel/2024-February/061760.html>

 **Fix Me!**

 **Fix Me!**

Workaround für Gruppenrechte:

<https://docs.software-univention.de/keycloak-app/latest/configuration.html#restrict-access-to-applications>

 **Fix Me!**

From: <https://wiki.deepdoc.at/dokuwiki/> - DEEPDOC.AT - enjoy your brain

Permanent link: https://wiki.deepdoc.at/dokuwiki/doku.php?id=virtualisierung:proxmox_kvm_und_lxc:openid_keycloak_anbindung_proxmox

Last update: 2025/11/29 22:06

