LDAP INTEGRATION ON KEYCLOAK

Keycloak can retrieve users from LDAP, synchronize groups, roles or custom attributes. Let's have a complete tour of what you can do with this connector.

LDAP integration on Keycloak

Intro : Active Directory or LDAP ?

LDAP (Lightweight Directory Access Protocol) is a protocol for directory service providers. Active Directory is a directory service provider. LDAP is the "API", Active Directory the "backend" with the database.

Microsoft reveal Active Directory in 1996, first wide deployment was in 1999 with Windows 2000 server

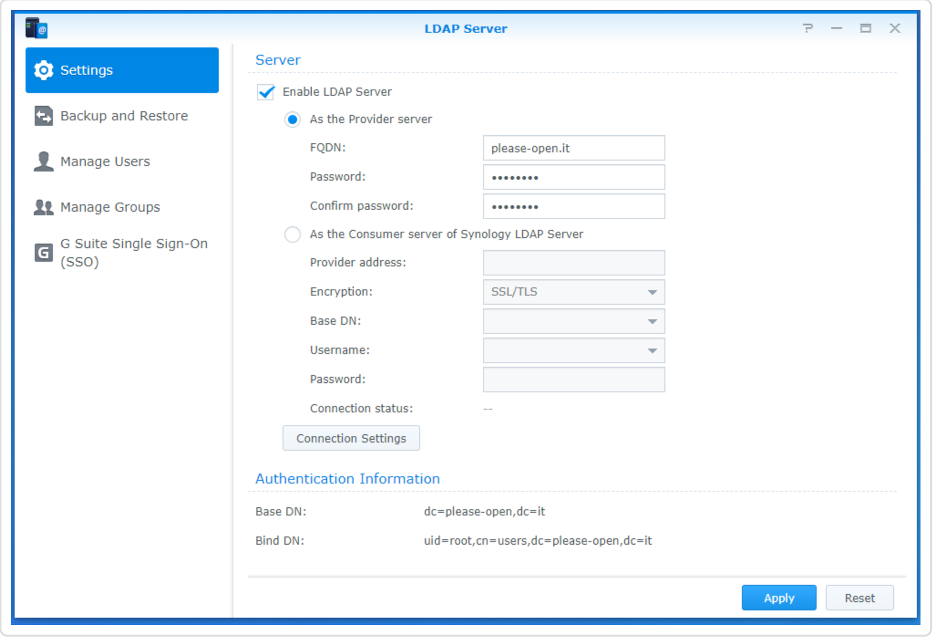
LDAP was created in 1992, that's why this protocol is mostly used on multi-users systems.

- https://ldapwiki.com/wiki/History%20of%20LDAP
- https://www.ajsnetworking.com/microsofts-active-directory/

Requirements

For this examples, we have :

- A running LDAP server with a directory behind. In our case, we use a Synology LDAP Server. All examples can run with any LDAP server, depending on your system. Of course, Windows Server Active Directory works perfectly
- A Keycloak realm. Feel free to subscribe to our Keycloak As A Service. All examples were made with a free realm.
- LDAP Admin, a free software for LDAP directory management. We will use only this interface.



Server

☑ Enable LDAP Server

● As the Provider server

FQDN: please-open.it
Password: ••••••••
Confirm password: ••••••••

○ As the Consumer server of Synology LDAP Server

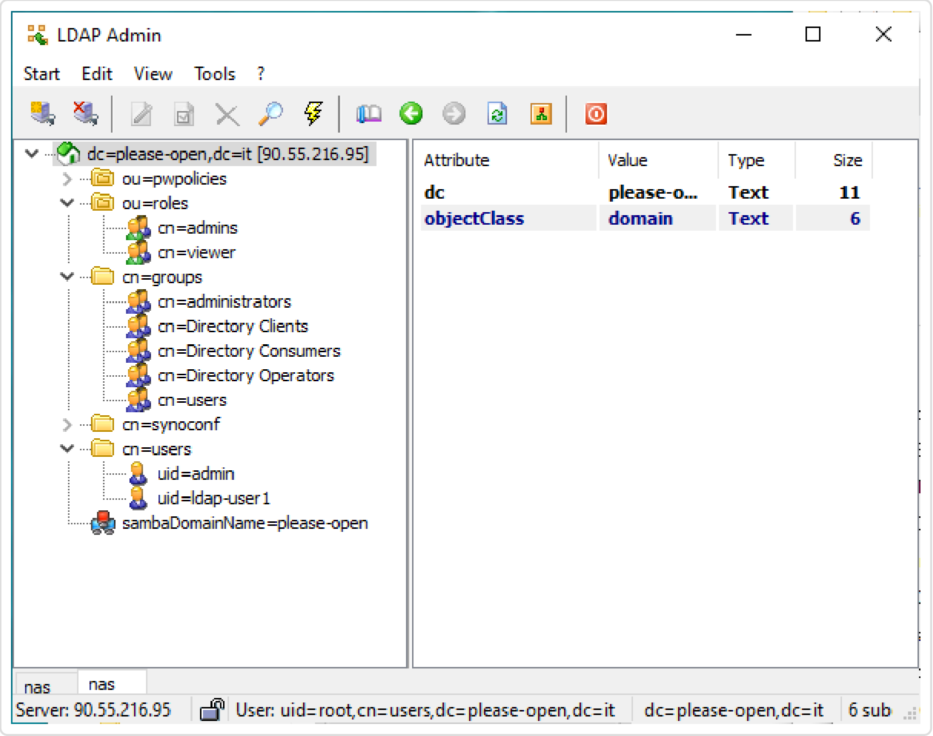
Provider address:
Encryption: SSL/TLS
Base DN:
Username:
Password:
Connection status: --

Authentication Information

Base DN: dc=please-open,dc=it
Bind DN: uid=root,cn=users,dc=please-open,dc=it

LDAP structure

LDAP Admin can show all details on any node of your directory. Very useful to use it when you need to integrate LDAP in other systems with complete references.



```
dc=please-open,dc=it [90.55.216.95]
  ou=pwpolicies
  ou=roles
    cn=admins
    cn=viewer
  cn=groups
    cn=administrators
    cn=Directory Clients
    cn=Directory Consumers
    cn=Directory Operators
    cn=users
  cn=synoconf
  cn=users
    uid=admin
    uid=ldap-user1
  sambaDomainName=please-open
```

Attribute	Value	Type	Size
dc	please-o...	Text	11
objectClass	domain	Text	6

Server: 90.55.216.95 User: uid=root,cn=users,dc=please-open,dc=it dc=please-open,dc=it 6 sub

Our directory has a standard structure :

- Roles (as Organization Unit) with groups
- Groups (as Common Name, as Synology created it) with groups
- List of users

Keycloak user federation VS identity provider

There are 2 entries in Keycloak : User federation and identity provider. Those two approaches are significantly different, by the method and the result. The result is not how users are provided.

Identity provider

By "identity provider", it means an identity server. Authentication process is delegated to the identity provider. Only if the user exists, is correctly authenticated and gives the right permissions the identity provider will return the identity. Check our blogpost about Authorization code.

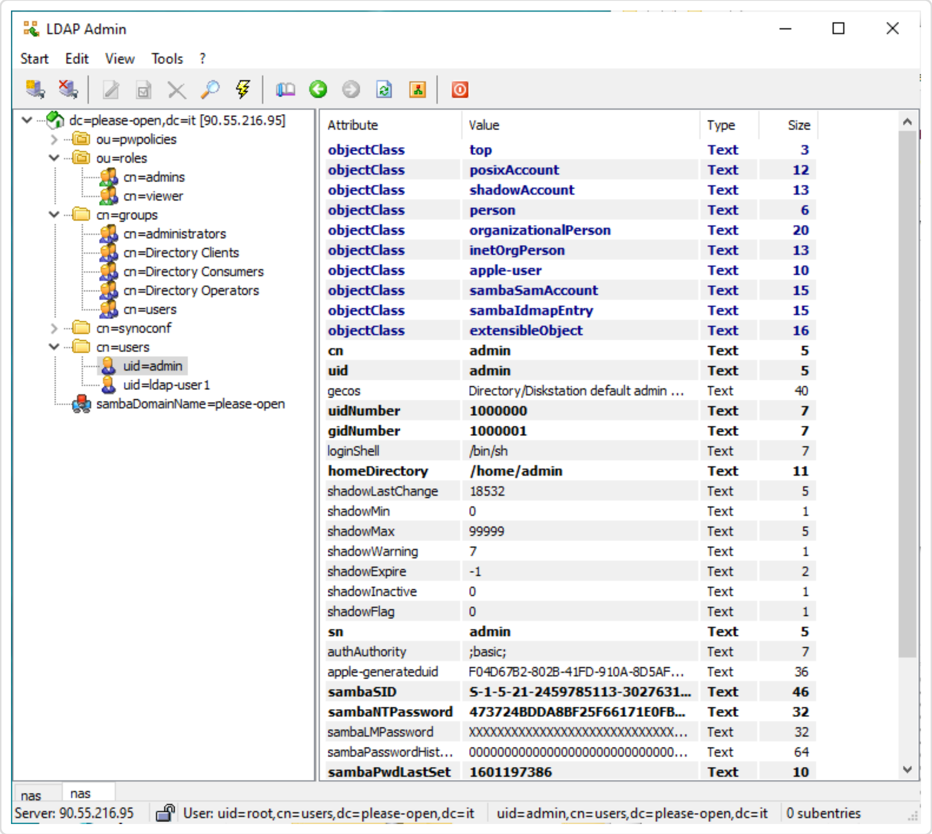
User federation

https://www.janua.fr/understanding-keycloak-user-federation/

"The way we store users" could be a good way to explain it. We can replace the user's management delivered by default in Keycloak by another method. It works with LDAP or Keberos by default, there are also some developer interfaces (SPI) if you want to code your own user management.

Add LDAP to Keycloak

First of all, we have to get the user structure and properties. With LDAP Admin it is easy :



dc=please-open,dc=it [90.55.216.95]
- ou=pwpolicies
- ou=roles
  - cn=admins
  - cn=viewer
- cn=groups
  - cn=administrators
  - cn=Directory Clients
  - cn=Directory Consumers
  - cn=Directory Operators
  - cn=users
- cn=synoconf
- cn=users
  - uid=admin
  - uid=ldap-user1
- sambaDomainName=please-open

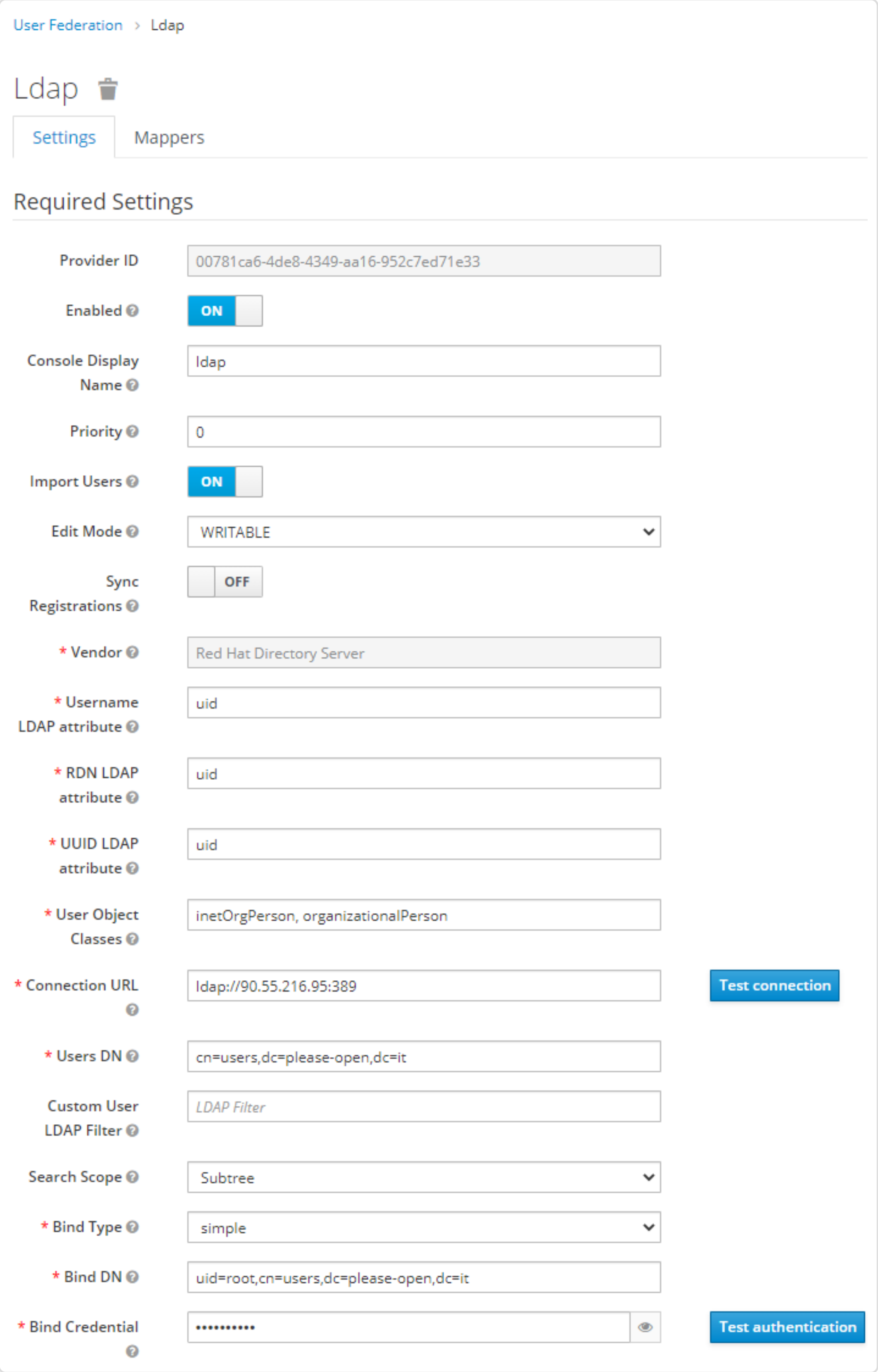
Attribute	Value	Type	Size
objectClass	top	Text	3
objectClass	posixAccount	Text	12
objectClass	shadowAccount	Text	13
objectClass	person	Text	6
objectClass	organizationalPerson	Text	20
objectClass	inetOrgPerson	Text	13
objectClass	apple-user	Text	10
objectClass	sambaSamAccount	Text	15
objectClass	sambaIdmapEntry	Text	15
objectClass	extensibleObject	Text	16
cn	admin	Text	5
uid	admin	Text	5
gecos	Directory/Diskstation default admin ...	Text	40
uidNumber	1000000	Text	7
gidNumber	1000001	Text	7
loginShell	/bin/sh	Text	7
homeDirectory	/home/admin	Text	11
shadowLastChange	18532	Text	5
shadowMin	0	Text	1
shadowMax	99999	Text	5
shadowWarning	7	Text	1
shadowExpire	-1	Text	2
shadowInactive	0	Text	1
shadowFlag	0	Text	1
sn	admin	Text	5
authAuthority	;basic;	Text	7
apple-generateduid	F04D67B2-802B-41FD-910A-8D5AF...	Text	36
sambaSID	S-1-5-21-2459785113-3027631...	Text	46
sambaNTPassword	473724BDDA8BF25F66171E0FB...	Text	32
sambaLMPassword	XXXXXXXXXXXXXXXXXXXXXXXXXXXX...	Text	32
sambaPasswordHist...	0000000000000000000000000000...	Text	64
sambaPwdLastSet	1601197386	Text	10

Server: 90.55.216.95 User: uid=root,cn=users,dc=please-open,dc=it uid=admin,cn=users,dc=please-open,dc=it 0 subentries

Then we have enough informations to fill the LDAP configuration in Keycloak.

- Username attribute is **uid**
- Two classes for user object : **inetOrgPerson and organizationalPerson**
- Users DN is **cn=users,dc=please-open,dc=it**
- Admin user is : **uid=root,cn=users,dc=please-open,dc=it**

Our advice is : do not rely on default values provided by "vendor" values. Check the right values using a LDAP explorer software.



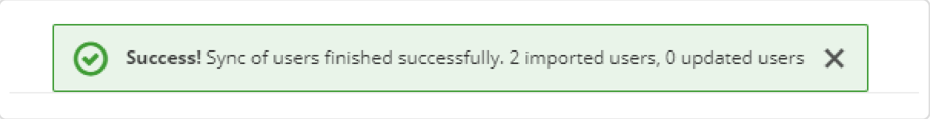
User Federation › Ldap

Ldap

Required Settings

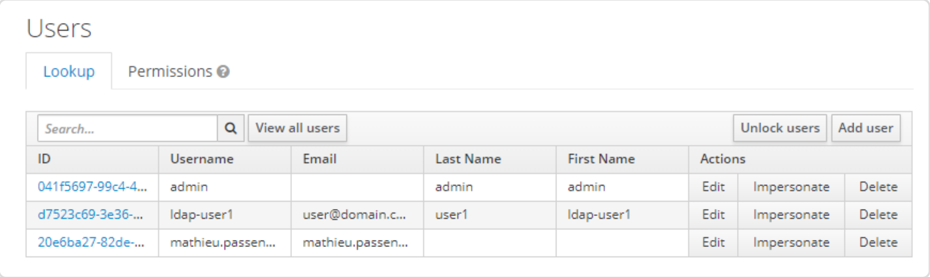
Provider ID	00781ca6-4de8-4349-aa16-952c7ed71e33
Enabled	ON
Console Display Name	ldap
Priority	0
Import Users	ON
Edit Mode	WRITABLE
Sync Registrations	OFF
* Vendor	Red Hat Directory Server
* Username LDAP attribute	uid
* RDN LDAP attribute	uid
* UUID LDAP attribute	uid
* User Object Classes	inetOrgPerson, organizationalPerson
* Connection URL	ldap://90.55.216.95:389
* Users DN	cn=users,dc=please-open,dc=it
Custom User LDAP Filter	LDAP Filter
Search Scope	Subtree
* Bind Type	simple
* Bind DN	uid=root,cn=users,dc=please-open,dc=it
* Bind Credential	••••••••••

Save the configuration, then you can synchronize all users for the first time.



Success! Sync of users finished successfully. 2 imported users, 0 updated users

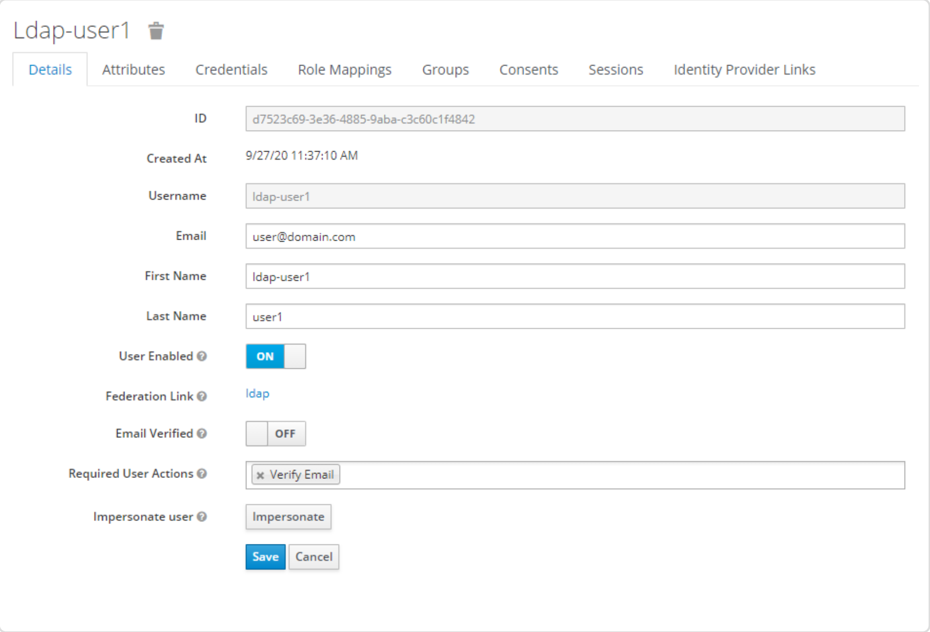
Our users are now in the database, let's take a look :



Users

ID	Username	Email	Last Name	First Name	Actions		
041f5697-99c4-4...	admin		admin	admin	Edit	Impersonate	Delete
d7523c69-3e36-...	ldap-user1	user@domain.c...	user1	ldap-user1	Edit	Impersonate	Delete
20e6ba27-82de-...	mathieu.passen...	mathieu.passen...			Edit	Impersonate	Delete

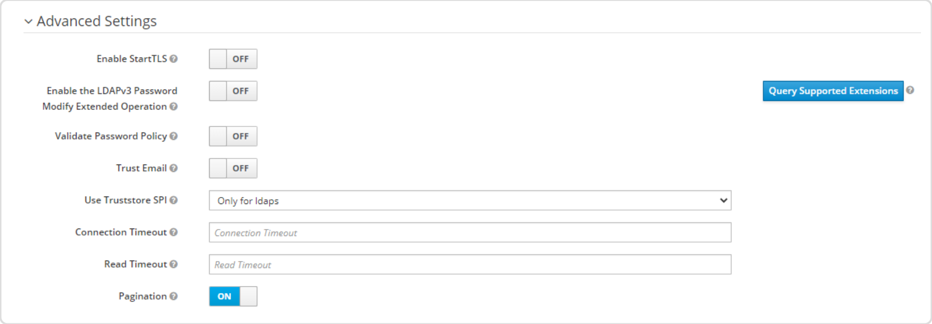
Users details are filled by the default import. Note the "Verify email" action required.



Ldap-user1

ID	d7523c69-3e36-4885-9aba-c3c60c1f4842
Created At	9/27/20 11:37:10 AM
Username	ldap-user1
Email	user@domain.com
First Name	ldap-user1
Last Name	user1
User Enabled	ON
Federation Link	ldap
Email Verified	OFF
Required User Actions	× Verify Email
Impersonate user	Impersonate

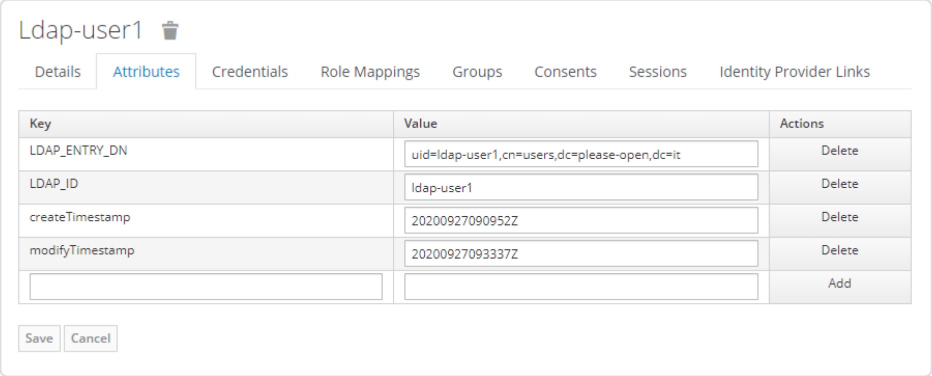
In the LDAP configuration we do not check "trust email" in "Advanced section"



Advanced Settings

Enable StartTLS	OFF
Enable the LDAPv3 Password Modify Extended Operation	OFF
Validate Password Policy	OFF
Trust Email	OFF
Use Truststore SPI	Only for ldaps
Connection Timeout	Connection Timeout
Read Timeout	Read Timeout
Pagination	ON

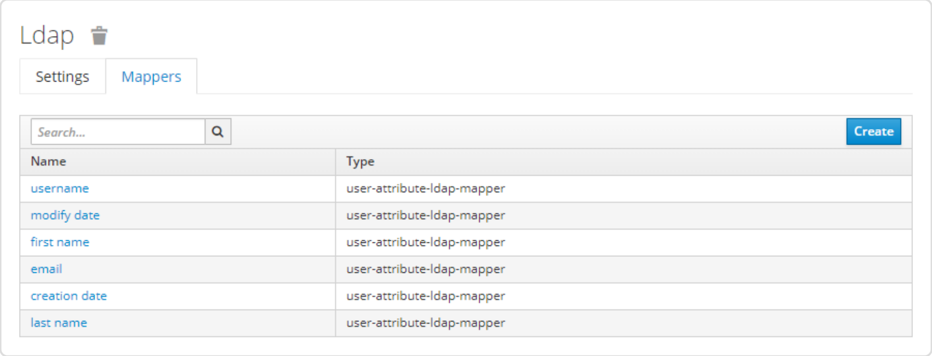
More attributes are also provided by default mappers :



Ldap-user1

Key	Value	Actions
LDAP_ENTRY_DN	uid=ldap-user1,cn=users,dc=please-open,dc=it	Delete
LDAP_ID	ldap-user1	Delete
createTimestamp	20200927090952Z	Delete
modifyTimestamp	20200927093337Z	Delete
		Add

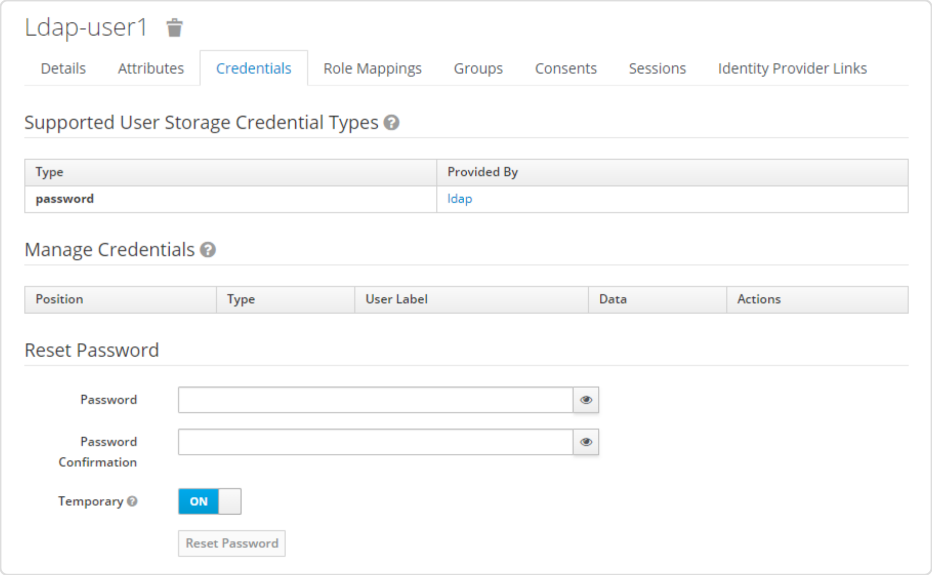
Those mappers are defined in the "mappers" section :



Ldap

Name	Type
username	user-attribute-ldap-mapper
modify date	user-attribute-ldap-mapper
first name	user-attribute-ldap-mapper
email	user-attribute-ldap-mapper
creation date	user-attribute-ldap-mapper
last name	user-attribute-ldap-mapper

Credentials are provided directly by the LDAP, so the entry shows this :



Ldap-user1

Supported User Storage Credential Types

Type	Provided By
password	ldap

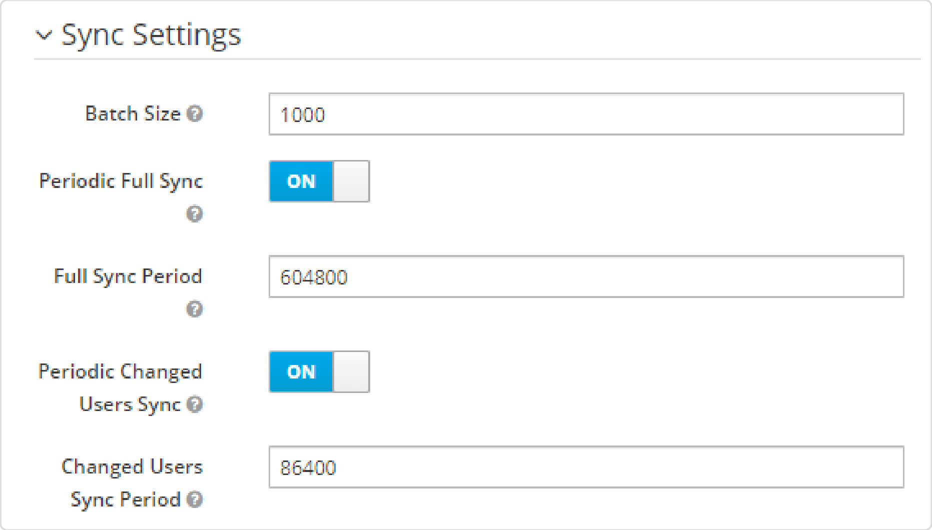
Manage Credentials

Position	Type	User Label	Data	Actions

Reset Password

Password	
Password Confirmation	
Temporary	ON

GREAT ! The user's synchronization works perfectly as we want. We define a synchronization period :



Sync Settings

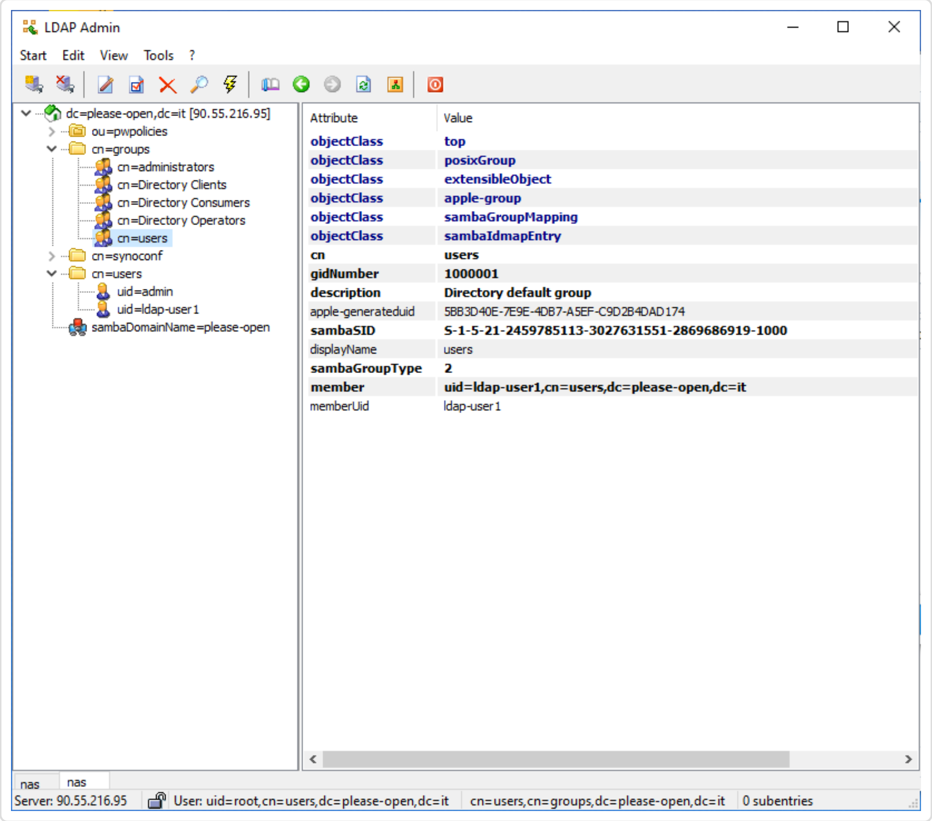
Batch Size	1000
Periodic Full Sync	ON
Full Sync Period	604800
Periodic Changed Users Sync	ON
Changed Users Sync Period	86400

Group Mapping

In LDAP, we can organize users by groups (with a property "memberOf"). This concept of "groups" does not exist in oauth2/openid Connect. There is only role mapping (realm roles or client roles) with users.

Keycloak has its own group implementation. This implementation is only a role mapping shortcut, each member of a group will have the role.

Check the groups structure in the LDAP Admin :



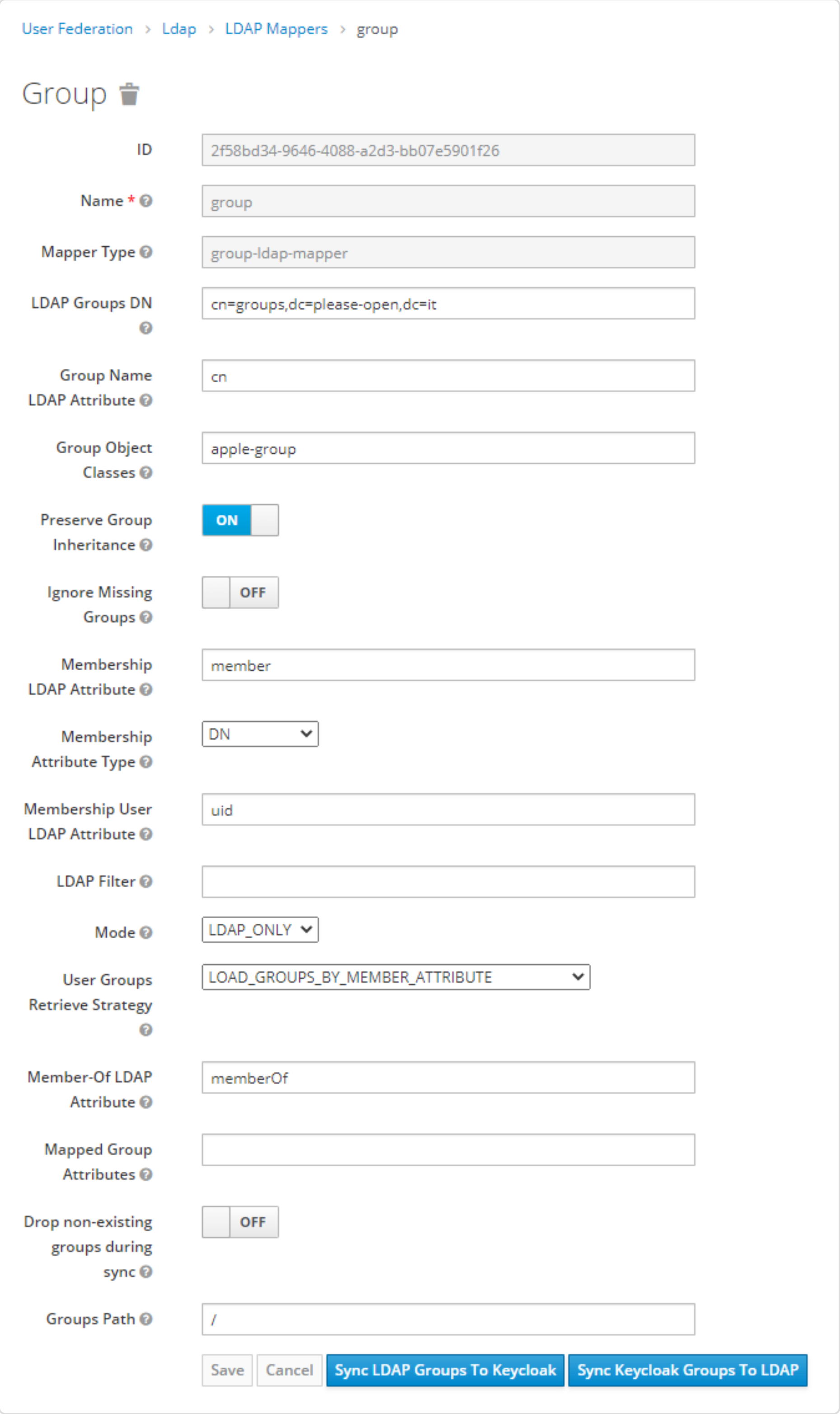
```
dc=please-open,dc=it [90.55.216.95]
  ou=pwpolicies
  cn=groups
    cn=administrators
    cn=Directory Clients
    cn=Directory Consumers
    cn=Directory Operators
    cn=users
  cn=synoconf
  cn=users
    uid=admin
    uid=ldap-user1
  sambaDomainName=please-open
```

Attribute	Value
objectClass	top
objectClass	posixGroup
objectClass	extensibleObject
objectClass	apple-group
objectClass	sambaGroupMapping
objectClass	sambaIdmapEntry
cn	users
gidNumber	1000001
description	Directory default group
apple-generateduid	5BB3D40E-7E9E-4DB7-A5EF-C9D2B4DAD174
sambaSID	S-1-5-21-2459785113-3027631551-2869686919-1000
displayName	users
sambaGroupType	2
member	uid=ldap-user1,cn=users,dc=please-open,dc=it
memberUid	ldap-user1

Server: 90.55.216.95 User: uid=root,cn=users,dc=please-open,dc=it cn=users,cn=groups,dc=please-open,dc=it 0 subentries

- LDAP Group DN is **cn=groups,dc=please-open,dc=it**
- Group object class is **apple-group**
- membership LDAP attribute is **member**
- User attribute is still **uid**

Now, add a new LDAP Mapper, with the type "group-ldap-mapper", then we have to fill a big form like seen previously :

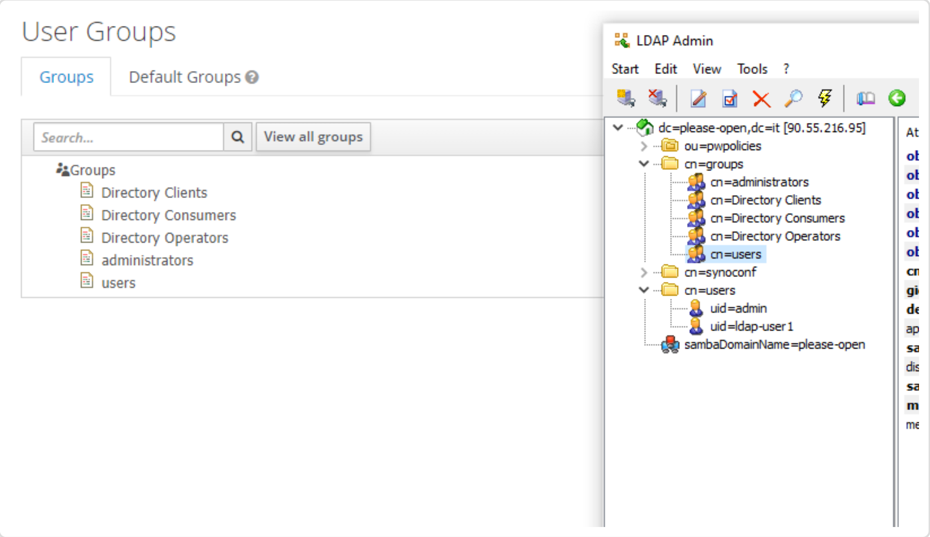


Group

ID	2f58bd34-9646-4088-a2d3-bb07e5901f26
Name *	group
Mapper Type	group-ldap-mapper
LDAP Groups DN	cn=groups,dc=please-open,dc=it
Group Name LDAP Attribute	cn
Group Object Classes	apple-group
Preserve Group Inheritance	ON
Ignore Missing Groups	OFF
Membership LDAP Attribute	member
Membership Attribute Type	DN
Membership User LDAP Attribute	uid
LDAP Filter	
Mode	LDAP_ONLY
User Groups Retrieve Strategy	LOAD_GROUPS_BY_MEMBER_ATTRIBUTE
Member-Of LDAP Attribute	memberOf
Mapped Group Attributes	
Drop non-existing groups during sync	OFF
Groups Path	/

Then, we sync all.

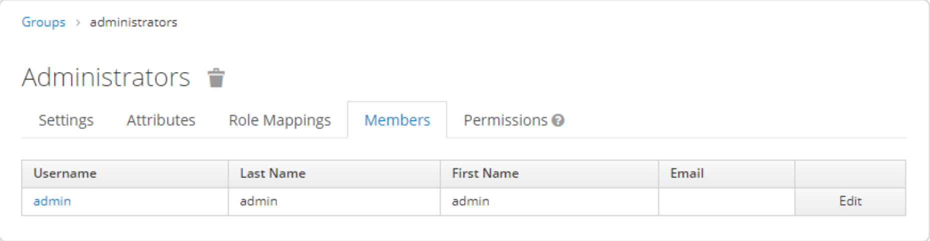
The Keycloak group structure is now the same that is defined in our directory :



```
Groups
  Directory Clients
  Directory Consumers
  Directory Operators
  administrators
  users
```

```
dc=please-open,dc=it [90.55.216.95]
  ou=pwpolicies
  cn=groups
    cn=administrators
    cn=Directory Clients
    cn=Directory Consumers
    cn=Directory Operators
    cn=users
  cn=synoconf
  cn=users
    uid=admin
    uid=ldap-user1
  sambaDomainName=please-open
```

Of course, we have correctly filled the users attributes so the users are affected to their groups.

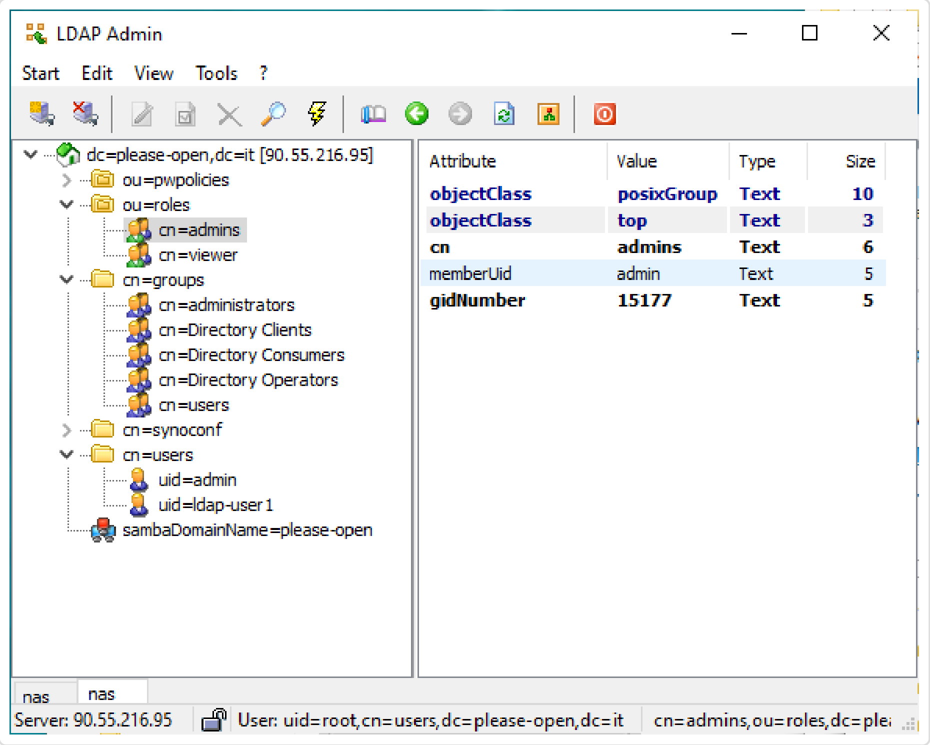


Administrators

Username	Last Name	First Name	Email	
admin	admin	admin		Edit

Role mapping

We have an Organizational Unit for roles. It is also the same as groups. In Keycloak, we will define the mapping directly to roles.



dc=please-open,dc=it [90.55.216.95]
- ou=pwpolicies
- ou=roles
  - cn=admins
  - cn=viewer
- cn=groups
  - cn=administrators
  - cn=Directory Clients
  - cn=Directory Consumers
  - cn=Directory Operators
  - cn=users
- cn=synoconf
- cn=users
  - uid=admin
  - uid=ldap-user1
- sambaDomainName=please-open

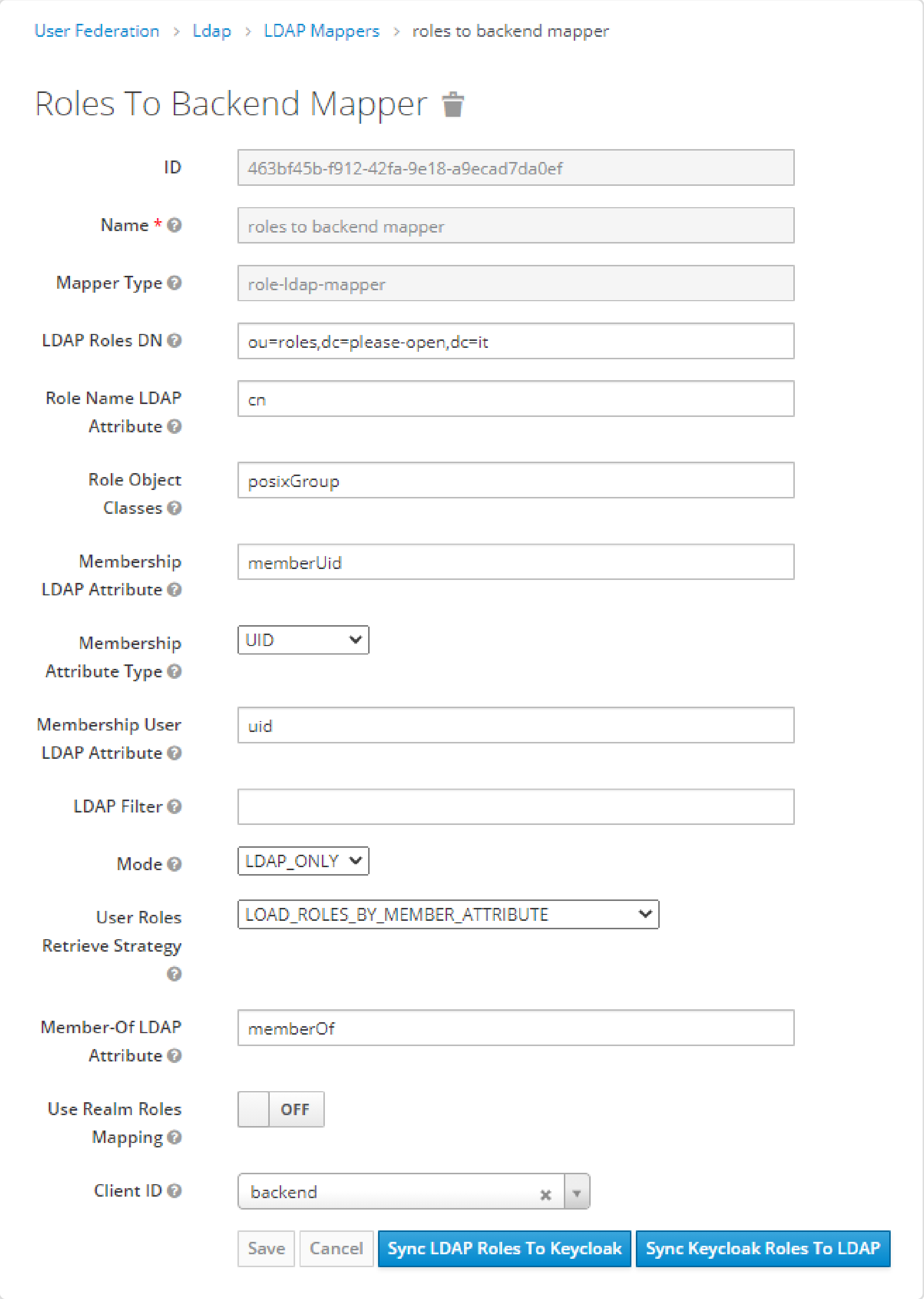
Attribute	Value	Type	Size
objectClass	posixGroup	Text	10
objectClass	top	Text	3
cn	admins	Text	6
memberUid	admin	Text	5
gidNumber	15177	Text	5

nas nas
Server: 90.55.216.95 User: uid=root,cn=users,dc=please-open,dc=it cn=admins,ou=roles,dc=plea

Check all attributes names :

- class is **posixGroup**
- name is **cn**
- Members defined in **memberUid**, directly with uid

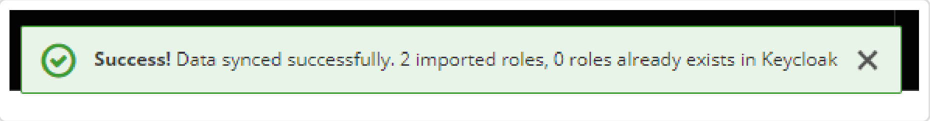
Define a new LDAP Mapper, using the mapper type "role-ldap-mapper". Fill all fields with values we got previously :


User Federation > Ldap > LDAP Mappers > roles to backend mapper
Roles To Backend Mapper

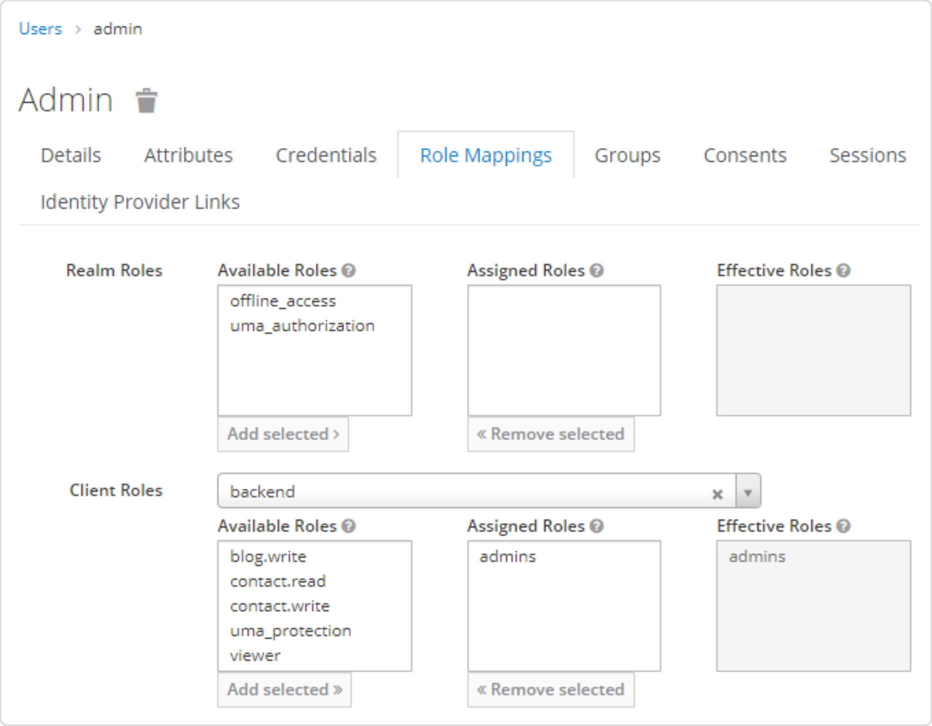
ID	463bf45b-f912-42fa-9e18-a9ecad7da0ef
Name *	roles to backend mapper
Mapper Type	role-ldap-mapper
LDAP Roles DN	ou=roles,dc=please-open,dc=it
Role Name LDAP Attribute	cn
Role Object Classes	posixGroup
Membership LDAP Attribute	memberUid
Membership Attribute Type	UID
Membership User LDAP Attribute	uid
LDAP Filter	
Mode	LDAP_ONLY
User Roles Retrieve Strategy	LOAD_ROLES_BY_MEMBER_ATTRIBUTE
Member-Of LDAP Attribute	memberOf
Use Realm Roles Mapping	OFF
Client ID	backend

2 ways exist for roles : realm role (global) or client based roles.

We chose the "backend" client for our use. The mapping will create roles in the client.


Success! Data synced successfully. 2 imported roles, 0 roles already exists in Keycloak

Users will have the roles assigned after synchronization :



Realm Roles
Available Roles: offline_access, uma_authorization
Assigned Roles:
Effective Roles:

Client Roles: backend
Available Roles: blog.write, contact.read, contact.write, uma_protection, viewer
Assigned Roles: admins
Effective Roles: admins

Keeping the same structure between LDAP and openid connect is great for already running projects. As consultants, we see many companies that rely on their directory structure, the only source of truth.

Keycloak can use the directory in read only mode or with a bi-directionnal synchronization. Do not forget to consider it.

Let's Get In Touch!

Any question ? Want more information ? Follow us on twitter or you can reach out to us via email.

contact@please-open.it

@PleaseOpen_It