univention
be open.

Migration guide: SimpleSAMLPHP to Keycloak

Univention GmbH

Mar 05, 2024

CONTENTS

CHAPTER

ONE

INTRODUCTION

This documentation is for system administrators who already operate UCS (Univention Corporate Server) 5.0 and explains the required steps for the migration from the single sign-on identity provider apps **SimpleSAMLphp** (SAML) and **OpenID Connect Provider** to the app **Keycloak**. The app **OpenID Connect Provider** uses **Kopano Konnect** to provide OpenID Connect capability to UCS.

1.1 Notes about UCS 5.2

Starting with UCS 5.2 the **Keycloak** app replaces the apps **SimpleSAMLphp** and **OpenID Connect Provider** as the default identity providers in UCS. The reason for this change is that **Keycloak** has many advantages in terms of features, configurability, and maintainability over the alternatives, for example, Keycloak provides OIDC and SAML endpoints in one component.

Warning: Migration from **SimpleSAMLphp** to **Keycloak** is mandatory before upgrading from UCS 5.0 to UCS 5.2. **SimpleSAMLphp** and **OpenID Connect Provider** are deprecated and will be removed in UCS 5.2.

If you use single sign-on for authentication in your UCS domain, read this document, migrate all services to use **Keycloak** as IdP (Identity Provider) and complete the migration with the steps in *Prepare for the update to UCS 5.2* (page 27).

If you are absolutely sure that single sign-on for authentication isn't used in your UCS domain, you can skip the migration part and just prepare your domain for the update to UCS 5.2, following the steps in *Prepare for the update to UCS 5.2* (page 27).

1.2 About this document

This document covers the following topics:

This documentation doesn't cover the following topics:

- Detailed information about the usage of the **Keycloak** app, see *Univention Keycloak app documentation* [1]
- Usage of UCS, see *UCS 5.0 Manual* [2].

To understand this documentation, you need to know the following concepts and tasks:

- Use and navigate in a remote shell on Debian GNU/Linux derivative Linux distributions like UCS. For more information, see Shell and Basic Commands[1] from *The Debian Administrator's Handbook*, Hertzog and Mas [3].
- Installation[2] and Configuration[3] of the app **`Keycloak`** as described in *Univention Keycloak app documentation* [1].
- Know the concepts of SAML (Security Assertion Markup Language[4]) and OIDC (OpenID Connect[5]) and the differences between the two standards.

Your feedback is welcome and highly appreciated. If you have comments, suggestions, or criticism, please send your feedback[6] for document improvement.

[1] https://www.debian.org/doc/manuals/debian-handbook/short-remedial-course.en.html#sect.shell-and-basic-commands
[2] https://docs.software-univention.de/keycloak-app/latest/installation.html#app-installation
[3] https://docs.software-univention.de/keycloak-app/latest/configuration.html#app-configuration
[4] https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language
[5] https://en.wikipedia.org/wiki/OpenID#OpenID_Connect_(OIDC)
[6] https://www.univention.com/feedback/?keycloak-migration=generic

CHAPTER

TWO

LIMITATIONS OF THE KEYCLOAK APP

The **Keycloak** app has not the same feature set as the **SimpleSAMLphp** integration at the time of writing. All of the following points are currently not supported by the **Keycloak** app, but will be implemented in the next versions.

For more information about limitations of the **Keycloak** app, refer to Requirements and limitations[7] in *Univention Keycloak app documentation* [1].

2.1 App authorization

In **SimpleSAMLphp** it's possible to restrict the access to certain clients through a checkbox on the user object in the UMC (Univention Management Console). To restrict the access of users to certain clients isn't possible with **Keycloak** at the moment.

2.2 Customization of the login page

With **SimpleSAMLphp** it's possible to adjust the login page through various UCR variables. You can use these UCR variables to add links to the sign-in dialog, for example to redirect the user to the User self services[8] in case they forgot their password.

With **Keycloak** you can adjust the theming of the sign-in dialog, but adding custom links isn't supported for the moment.

[7] https://docs.software-univention.de/keycloak-app/latest/limitations.html#app-limitations
[8] https://docs.software-univention.de/manual/5.0/en/user-management/user-self-service.html#user-management-password-changes-by-users

CHAPTER

THREE

MIGRATION PROCEDURE

Keycloak replaces **SimpleSAMLphp** and the app **OpenID Connect Provider** as SAML IDP[9] and OIDC provider[10] in a future release of UCS. This section provides a general overview of the migration steps and the required considerations to make before migrating. This migration guide focuses on exclusively on UCS 5.0.

Before the migration can take place, please keep in mind:

- You can migrate services step by step.
- The migration is a manual process.
- Create a backup of the current single sign-on configuration of your services **before** the migration, so that you can rollback in case a problem occurs.
- **SimpleSAMLphp** and **OpenID Connect Provider** still work even if you installed **Keycloak**.
- After you migrated a service, existing user sessions become invalid. Users have to sign in to the migrated service again.

The migration of one or multiple services always includes at least the following steps:

- The installation and configuration of the **Keycloak** app. For a detailed description, see Installation[11] in the *Univention Keycloak app documentation* [1].
- The creation of OIDC RP[12] or SAML SP[13], referred to as *clients* in this document, in **Keycloak** for each service. For more information about how to create those clients, refer to *Migration of services using OIDC for authentication* (page 6) and *Migration of services using SAML for authentication* (page 7).
- The update of the single sign-on configuration of the services to use **Keycloak** as IdP. Have a look at the examples in *Samples for migration of SAML and OIDC services* (page 17).
- The verification that single sign-on works with **Keycloak** as IdP

3.1 Prerequisites

Before the migration can take place, verify to comply to the following prerequisites:

- At least one service uses single sign-on.
- The UCS Primary Directory Node is on UCS version 5.0-4 with the latest errata updates.
- The servers where you want to install **Keycloak** must also be on UCS version 5.0-4 with the latest errata updates.
- If you have **Keycloak** already installed, make sure to update to the latest app version.

[9] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[10] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[11] https://docs.software-univention.de/keycloak-app/latest/installation.html#app-installation
[12] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-RP
[13] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

3.2 Migration of services using OIDC for authentication

This section gives a general idea about the migration of services that use **OpenID Connect Provider** as OIDC Provider[14] for the authentication to **Keycloak** as OIDC Provider[15].

The general approach for the migration includes the following:

- Install the latest version of the **Keycloak** app in the UCS domain.
- Get an overview of all the services that use **OpenID Connect Provider** and their settings.
- Create an OIDC RP[16], the client, in **Keycloak** for every service that uses **OpenID Connect Provider** as OIDC Provider[17].
- Change the OIDC (OpenID Connect) settings in the services to use **Keycloak** as OIDC Provider[18] and validate the setup.

To setup a service for OIDC with **Keycloak** use the following steps:

1. The UDM (Univention Directory Manager) module `oidc/rpservice` configures services that use **OpenID Connect Provider**. To get a list of all the services and settings, run the following command on the UCS *Primary Directory Node*:

   Listing 3.1: List all services that use *OpenID Connect Provider* for OIDC

   ```
   $ udm oidc/rpservice list
   ```

2. Each service has a `clientid`, a `clientsecret`, and a `redirectURI`. You need the values of these settings to create identical clients for the service in **Keycloak**. On the UCS system where you have installed **Keycloak**, create an OIDC client with the following command:

   Listing 3.2: Create an *OIDC client* for the service in *Keycloak*

   ```
   $ univention-keycloak oidc/rp create \
     --client-secret clientsecret \
     --app-url redirectURI \
     clientid
   ```

   Note: In case you made custom settings of your **OpenID Connect Provider** installation, review the following files on your UCS system, that has the app installed:

   - `/etc/kopano/konnectd.cfg`
   - `/etc/kopano/konnectd-identifier-registration.yaml`
   - `/etc/kopano/konnectd-identifier-scopes.yaml`

3. You can also use the Keycloak Admin Console[19] to create OIDC clients manually or to adjust clients created with `univention-keycloak oidc/rp create`. See also Keycloak as OpenID Connect provider[20] for more information on how to manage OIDC client clients with **Keycloak**.

4. After you created the OIDC client for your service, you need to change the IdP settings that point to the OIDC Provider[21] in the OIDC configuration of the service. Because the services are highly individual in the way they configure OIDC, this documentation can't provide a general description. At least, you need the base URL of your **Keycloak** server. Run the following command on the UCS system that has **Keycloak** installed:

[14] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[15] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[16] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-RP
[17] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[18] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[19] https://docs.software-univention.de/keycloak-app/latest/installation.html#keycloak-admin-console
[20] https://docs.software-univention.de/keycloak-app/latest/configuration.html#oidc-op
[21] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider

Listing 3.3: Get base URL of the *Keycloak* server

```
$ univention-keycloak get-keycloak-base-url
```

5. Some services may need to configure the OIDC Provider[22] URL. Its value is *SSO_URL*/realms/ucs. Replace *SSO_URL* with the output from the previous command.

   Other services may have individual settings for the authorization endpoint, the token endpoint, and so on. To get these URLs, run the following commands on the **Keycloak** server.

   Listing 3.4: Get different endpoint URLs

```
$ SSO_URL="$(univention-keycloak get-keycloak-base-url)"
$ univention-install jq
$ curl "$SSO_URL/realms/ucs/.well-known/openid-configuration" | jq
{
  "issuer": "https://ucs-sso-ng.example.com/realms/ucs",
  "authorization_endpoint": "https://ucs-sso-ng.example.com/realms/ucs/
↪protocol/openid-connect/auth",
  "token_endpoint": "https://ucs-sso-ng.example.com/realms/ucs/protocol/openid-
↪connect/token",
  "introspection_endpoint": "https://ucs-sso-ng.example.com/realms/ucs/
↪protocol/openid-connect/token/introspect",
  "userinfo_endpoint": "https://ucs-sso-ng.example.com/realms/ucs/protocol/
↪openid-connect/userinfo",
  "end_session_endpoint": "https://ucs-sso-ng.example.com/realms/ucs/protocol/
↪openid-connect/logout",
  ...
}
```

   You don't need to change the settings for the client name and secret, because you have created an OIDC client with identical values in **Keycloak**.

To get a better picture using OIDC with **Keycloak**, have a look at the examples given in section *Services using OIDC* (page 23).

3.3 Migration of services using SAML for authentication

This section gives a general idea about the migration of services that use **SimpleSAMLphp** for the authentication to **Keycloak** as a SAML client.

The general approach for the migration includes the following:

- Install the latest version of the **Keycloak** app in the UCS domain.
- Get an overview of all the services that use **SimpleSAMLphp** and their settings.
- Check and create attribute mappers for LDAP. Selected LDAP attributes become available to **Keycloak**.
- Create a SAML SP[23], the client, and necessary attribute mappers in **Keycloak** for every service that uses **SimpleSAMLphp** as SAML IDP[24].
- Change the SAML (Security Assertion Markup Language) settings in the services to use **Keycloak** as a SAML IDP[25] and validate the setup.

The following sections explain each step of the migration in detail.

[22] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[23] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[24] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[25] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP

3.3.1 Create LDAP attribute mappers

Some services require specific LDAP attributes, for example, the service requires the user's email address, which the SAML IDP[26] can pass to the service during authentication. Create appropriate LDAP attribute mappers based on your needs.

The user object in **Keycloak** doesn't have every attribute available in the LDAP directory. You can find a list of attributes mapped to **Keycloak** in Import of user attributes from UCS to Keycloak[27] of *Univention Keycloak app documentation* [1].

To check, if the attribute mapping is sufficient for your services, compare the list with the attributes that **SimpleSAMLphp** reads from the LDAP directory. To get a complete list of LDAP attributes available to **SimpleSAMLphp**, run the following command:

Listing 3.5: List IDP LDAP attributes

```
$ udm saml/idpconfig list | sed -n 's/LdapGetAttributes: //p'
```

If you find attributes, that don't map to Keycloak, you can create an LDAP mapper object through the tool **univention-keycloak** or directly in the Keycloak Admin Console[28]. To create a mapping on the command line, run the following command:

Listing 3.6: Create LDAP attribute mapping

```
$ univention-keycloak user-attribute-ldap-mapper create $LDAP_ATTRIBUTE_NAME
```

3.3.2 Create a SAML client in Keycloak

Each SAML SP[29], or client, that you have configured for **SimpleSAMLphp**, requires a corresponding SAML SP[30] created in **Keycloak** with the appropriate settings. Depending on the specific settings of your SAML SP[31], there are several ways to create a SAML client:

To inspect the **SimpleSAMLphp** configuration on the command line, run:

Listing 3.7: List **SimpleSAMLphp** Provider configuration

```
$ udm saml/serviceprovider list
```

Through the UMC module *SAML Identity Provider*. For a description of each LDAP attribute in this object, see Adding a new external service provider[32] in *UCS 5.0 Manual* [2].

Use one of the following approaches that best suits your needs:

1. If your application provides a metadata XML file for the SAML client settings, you can create a client in **Keycloak** with the command line tool **univention-keycloak** by using either the URL to that XML file or the file itself.

Listing 3.8: Create basic Keycloak client

```
# with URL
$ univention-keycloak saml/sp create --metadata-url="$URL"

# with local XML file
$ univention-keycloak saml/sp create \
```

(continues on next page)

[26] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[27] https://docs.software-univention.de/keycloak-app/latest/configuration.html#ldap-attribute-mapper
[28] https://docs.software-univention.de/keycloak-app/latest/installation.html#keycloak-admin-console
[29] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[30] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[31] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[32] https://docs.software-univention.de/manual/5.0/en/domain-ldap/saml.html#domain-saml-additional-serviceprovider

(continued from previous page)

```
--metadata-url="REPLACE WITH SAML CLIENT ISSUER NAME OR URL" \
--metadata-file "/path/to/xml"
```

2. You can transfer each of the SAML client settings from the `saml/serviceprovider` object to the **Keycloak** client. The tool **univention-keycloak** has flags and options for each of these settings.

Table 3.1: Mapping between **SimpleSAMPLphp** settings and **univention-keycloak saml/sp create** options

UMC SAML Identity Provider module / UDM attribute name	Keycloak GUI	`univention-keycloak saml/sp create` option
Service provider activation status: `isActivated`	Enabled	`--not-enabled`
Service provider identifier: `Identifier`	Client ID	`--client-id`
Respond to this service provider URL after login: `AssertionConsumerService`	Valid redirect URI. This value is also automatically parsed from a metadata XML file.	`--valid-redirect-uris`
Format of *NameID* attribute: `NameIDFormat`	Name ID format	`--name-id-format`
Description of this service provider: `serviceproviderdescription`	Description	`--description`
URL to the service provider's privacy policy: `privacypolicyURL`	Policy URI	`--policy-url`
Respond to this service provider URL after login: `AssertionConsumerService`	Assertion Consumer Service POST Binding URL	`--assertion-consumer-url-post`
Single logout URL for this service provider: `singleLogoutService`	Logout Service POST Binding URL	`--single-logout-service-url-post`
Allow transition of LDAP attributes to the service provider: `simplesamlAttributes`	n/a	Not available in Keycloak, only implicitly configured if specific attributes are configured to be transitioned to the service provider.
Name of the attribute that is used as NameID: `simplesamlNameIDAttribute`	n/a	Not available during the creation of the SAML SP[33] client. Instead another mappings object has to be created for **Keycloak**, see *Configure user attributes for SAML response* (page 10).
List of LDAP attributes to transmit: `LDAPattributes`	n/a	Not available during the creation of the SAML SP[34] client. Instead another mappings object has to be created for **Keycloak**, see *Configure user attributes for SAML response* (page 10).
Value for attribute format field: `attributesNameFormat`	n/a	Not available during the creation of the SAML SP[35] client. Instead another mappings object has to be created for **Keycloak**, see *Configure user attributes for SAML response* (page 10).

[33] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[34] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[35] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

3. You can also use the Keycloak Admin Console[36] to create SAML clients manually or to adjust clients created with `univention-keycloak saml/sp create`. You may want to consult the **Keycloak** documentation at *Managing OpenID Connect and SAML Clients* [4] about how to create clients in Keycloak.
4. Some SAML clients may use custom configurations (e.g. with the attribute `rawsimplesamlSPconfig` of the `saml/serviceprovider` object). If you have a service configured in that way, please check the `/etc/simplesamlphp/metadata.d/*.php` files to get the appropriate settings for the client.

3.3.3 Configure user attributes for SAML response

The following listing demonstrates how to create attribute mappings in **Keycloak** for the `simplesamlNameIDAttribute` and `LDAPattributes` settings of the `saml/serviceprovider` object. These mappings allow you to configure the unique identifier for the user and additional user attributes for the SAML response.

NameID attribute (`simplesamlNameIDAttribute`)
To map the attribute to **Keycloak** that uniquely identifies a user, create a so-called `SAML client nameid mapper` and attach it to the SAML SP[37]. The table Table 3.2 shows which **SimpleSAMLphp** settings correspond to which options of the command `univention-keycloak saml-client-nameid-mapper create`.

Table 3.2: Mapping between **SimpleSAMPLphp** settings and **`univention-keycloak saml-client-nameid-mapper create`** options

UMC SAML Identity Provider module / UDM attribute name	Keycloak GUI	`univention-keycloak saml-client-nameid-mapper create` option
Name of the attribute used as *NameID*: `simplesamlNameIDAttribute`	User attribute	`--user-attribute`
Format of *NameID* attribute: `NameID-Format`	Mapper *NameID* format	`--mapper-nameid-format`

For example, a command might look like one of the following:

[36] https://docs.software-univention.de/keycloak-app/latest/installation.html#keycloak-admin-console
[37] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

Listing 3.9: Create `client nameid mapper`

```
$ univention-keycloak saml-client-nameid-mapper create CLIENTID MAPPERNAME \
  --user-attribute uid \
  --mapper-nameid-format urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
```

Additional user attributes (`LDAPattributes`)

Some SAML SP[38] require additional attributes. The name of the attribute on the user object requires a mapping to the name of the attribute in the SAML assertion. The table Table 3.3 shows which **`SimpleSAMLphp`** settings correspond to which `univention-keycloak saml-client-user-attribute-mapper create` options.

Table 3.3: Mapping between **`SimpleSAMLphp`** settings and **`univention-keycloak saml-client-user-attribute-mapper create`** options

UMC SAML Identity Provider module / UDM attribute name	Keycloak GUI	`univention-keycloak saml-client-nameid-mapper create` option
Value for attribute format field: `attributesNameFormat`	Name ID	`--attribute-nameformat`
List of LDAP attributes to transmit: `LDAPattributes`	User attribute, SAML attribute name	`--user-attribute --attribute-name`

You need to create such a mapper for each attribute listed in the SAML SP[39] configuration. In **`Keycloak`**, you find flags to specify the name of the attribute to send:

`--user-attribute`
: is the attribute name present on the **`Keycloak`** user object.

`--attribute-name`
: is the SAML attribute utilizing the `urn:oid` namespace.

For example, a command might look like one of the following:

Listing 3.10: Create SAML client user attribute mapper

```
$ univention-keycloak saml-client-user-attribute-mapper create CLIENTID␣
↪MAPPERNAME \
  --user-attribute uid \
  --attribute-name urn:oid:0.9.2342.19200300.100.1.1
```

Note: To get a list of client IDs for all existing SAML SP[40] clients **`Keycloak`**, run:

[38] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[39] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[40] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

Listing 3.11: Get a list of client IDs for all existing SAML SP clients

```
$ univention-keycloak saml/sp get
```

3.3.4 Configure SAML services to use Keycloak

After creating SAML SP[41] clients and mappers in **Keycloak**, the next step is to update the service settings to use **Keycloak** as IdP.

1. Change the SAML settings in the services to use **Keycloak** as SAML IDP[42] and validate the setup. The service needs some information about **Keycloak** as the IdP. This includes the *Identity Provider Metadata URL*, and the public certificate.

   The SAML Identity Provider Metadata endpoint for Keycloak is `https://$sso_url/realms/ucs/protocol/saml/descriptor`. You can obtain *sso_url* from the output of the following command:

   Listing 3.12: Lookup Keycloak base URL

   ```
   $ sso_url="$(univention-keycloak get-keycloak-base-url)"
   ```

2. It's necessary to update the IdP public certificate in your SAML settings. You can obtain the **Keycloak** server certificate in PEM format with the following command:

   Listing 3.13: Retrieve public certificate and *Keycloak* base URL

   ```
   $ univention-keycloak saml/idp/cert get \
      --as-pem \
      --output "/tmp/keycloak.cert"
   ```

How and where you need to configure this information in the service itself is very specific to the service, and this document can't describe it in general terms. Consult the documentation of the service itself. To get a better idea of using SAML with **Keycloak**, take a look at the *Services using SAML* (page 17) in the next section.

3.3.5 UCR variable differences

This section describes the differences between UCR variables when using **SimpleSAMLphp** (SAML) and **Keycloak**.

Added variables

Keycloak introduces the following UCR variables:

- `keycloak/apache/config`[43]
- `keycloak/server/sso/path`[44]

[41] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP
[42] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[43] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-apache-config
[44] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-server-sso-path

Renamed variables

Keycloak uses the following variables known to be used for SAML before:

Table 3.4: UCR variable differences

SAML	Keycloak
ucs/server/sso/fqdn	keycloak/server/sso/fqdn[45]
ucs/server/sso/autoregistraton	keycloak/server/sso/autoregistration[46]
ucs/server/sso/virtualhost	keycloak/server/sso/virtualhost[47]
ucs/server/sso/password/change/server	keycloak/password/change/endpoint[48]
saml/apache2/ssl/ca	keycloak/apache2/ssl/ca[49]
saml/apache2/ssl/key	keycloak/apache2/ssl/key[50]
saml/apache2/ssl/certificate	keycloak/apache2/ssl/certificate[51]
saml/apache2/content-security-policy/.*	keycloak/csp/frame-ancestors[52]
saml/idp/selfservice/check_email_verification[53]	ucs/self/registration/check_email_verification[54]
saml/idp/log/level	keycloak/log/level[55]
saml/idp/selfservice/account-verification/error-title[56] and saml/idp/selfservice/account-verification/error-title/.*	keycloak/login/messages/en/accountNotVerifiedMsg[57] and keycloak/login/messages/de/accountNotVerifiedMsg[58]
saml/idp/selfservice/account-verification/error-descr[59] and saml/idp/selfservice/account-verification/error-descr/.*	keycloak/login/messages/en/accessDeniedMsg[60] and keycloak/login/messages/de/accessDeniedMsg[61]

[45] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-server-sso-fqdn
[46] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-server-sso-autoregistration
[47] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-server-sso-virtualhost
[48] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-password-change-endpoint
[49] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-apache2-ssl-ca
[50] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-apache2-ssl-key
[51] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-apache2-ssl-certificate
[52] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-csp-frame-ancestors
[53] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-saml-idp-selfservice-check_email_verification
[54] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-ucs-self-registration-check_email_verification
[55] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-log-level
[56] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-saml-idp-selfservice-account-verification-error-title
[57] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-login-messages-en-accountNotVerifiedMsg
[58] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-login-messages-de-accountNotVerifiedMsg
[59] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-saml-idp-selfservice-account-verification-error-descr
[60] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-login-messages-en-accessDeniedMsg
[61] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-login-messages-de-accessDeniedMsg

No longer supported variables

Keycloak doesn't support the following UCR variables anymore:

- saml/idp/authsource[62]
- saml/idp/ldap/debug
- saml/idp/ldap/get_attribute
- saml/idp/ldap/user
- saml/idp/log/debug/enabled
- saml/idp/negotiate/filter-subnets[63]
- saml/idp/session-duration
- saml/idp/show-error-reporting
- saml/idp/show-errors
- saml/idp/technicalcontactemail
- saml/idp/technicalcontactname
- saml/idp/timezone
- ucs/server/sso/certificate/download
- ucs/server/sso/certificate/generation

Keycloak *enables* the following UCR variables with the LDAP federation:

- saml/idp/ldap/enable_tls
- saml/idp/negotiate starting with **Keycloak** app version 22.0.3-ucs1.

Keycloak *sets* the following UCR variables with the LDAP federation:

- saml/idp/ldap/user
- saml/idp/ldap/search_attributes

Not used anymore

Keycloak doesn't use the following UCR variables anymore and automatically sets a respective configuration:

- saml/apache2/ssl/certificatechain
- saml/idp/certificate/certificate
- saml/idp/certificate/privatekey
- saml/idp/https

Keycloak redirects to HTTPS automatically.

For cookies, the following UCR variables existed:

- saml/idp/session-cookie/secure
- saml/idp/session-cookie/samesite
- saml/idp/language-cookie/secure
- saml/idp/language-cookie/samesite

[62] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-saml-idp-authsource
[63] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-saml-idp-negotiate-filter-subnets

To set the cookie policy for **Keycloak** use the UCR variable `keycloak/cookies/samesite`[64]. For possible values, see *Univention Keycloak app documentation* [1].

SAML doesn't use the following UCR variables anymore:

- `stunnel/debuglevel`
- `saml/idp/lookandfeel/theme`

To set the theme in **Keycloak**, use the command **`ucr set ucs/web/theme=dark|light`** for `ucs/web/theme`[65].

Use **`univention-keycloak saml`** to handle the SAML integration done with the following UCR variables before:

- `saml/idp/enableSAML20-IdP`
- `saml/idp/entityID`
- `saml/idp/entityID/supplement/.*`

3.4 Forward from legacy SimpleSAMLphp to Keycloak

This section is about single sign-on between services that already use **Keycloak** and other services that still use **SimpleSAMLphp**. This configuration can be a temporary solution for environments which have a lot of services to migrate, and where single sign-on between all services needs to be available during the time of the migration. Keep in mind, that the outlined setup is only a short-term solution for your environment until all clients completed the migration to **Keycloak**.

Important: This setup is only important if single sign-on between migrated and not migrated services is needed during the time of the migration. Future releases will not support this setup.

---

1. Download the **Keycloak** IdP's signing certificate for SAML communication and save it to the local file `/etc/ssl/certs/ucs-sso-ng.keycloak-signing.pem`:

Listing 3.14: Download **Keycloak** certificate

```
$ univention-keycloak saml/idp/cert get \
--as-pem \
--output /etc/ssl/certs/ucs-sso-ng.keycloak-signing.pem
```

2. Create a client for **SimpleSAMLphp** in **Keycloak**:

Listing 3.15: Create client for **SimpleSAMLphp**

```
$ univention-keycloak saml/sp create \
--umc-uid-mapper \
--metadata-url \
"https://${ucs_sso_fqdn}/simplesamlphp/module.php/saml/sp/metadata.php/default-
↪sp" \
--valid-redirect-uris \
"https://${ucs_sso_fqdn}/simplesamlphp/module.php/saml/sp/saml2-acs.php/
↪default-sp"
```

3. Change the default provider from `univention-ldap` to `default-sp`:

[64] https://docs.software-univention.de/keycloak-app/latest/configuration.html#envvar-keycloak-cookies-samesite
[65] https://docs.software-univention.de/manual/5.0/en/appendix/variables.html#envvar-ucs-web-theme

Listing 3.16: Change default provider

```
$ ucr set saml/idp/authsource=default-sp
```

4. Create a remote IdP for **Keycloak** in **SimpleSAMLphp**:

Listing 3.17: Create remote IdP for **Keycloak** in **SimpleSAMLphp**

```
$ kc_provider=$(univention-keycloak get-keycloak-base-url)
$ cat <<EOF > /etc/simplesamlphp/metadata/saml20-idp-remote.php
  <?php
  \$metadata['https://${kc_provider}/realms/ucs'] = [
    'SingleSignOnService'  => '${kc_provider}/realms/ucs/protocol/saml',
    'SingleLogoutService'  => '${kc_provider}/realms/ucs/protocol/saml',
    'certificate'          => 'ucs-sso-ng.keycloak-signing.pem',
    'authproc' => array(
      50 => array(
        'class' => 'core:AttributeCopy',
        'urn:oid:0.9.2342.19200300.100.1.1' => 'uid',
      ),
    ),
  ];
  EOF
```

CHAPTER

FOUR

SAMPLES FOR MIGRATION OF SAML AND OIDC SERVICES

This section provides detailed examples how to migrate SAML and OIDC services to **Keycloak**.

4.1 Services using SAML

The following examples demonstrate the migration of services that use SAML for authentication and **SimpleSAMLphp** as IdP to **Keycloak** as IdP.

4.1.1 UCS Portal

You can configure the UCS Portal to use SAML for authentication. For a detailed description of how to configure the UCS Portal to use **Keycloak** as a SAML IDP[66], refer to Use Keycloak for login to UCS Portal[67] in the *Univention Keycloak app documentation* [1].

4.1.2 Nextcloud

This section is about migrating the **Nextcloud** app to use **Keycloak** as SAML IDP[68] for the SAML authentication. It assumes that your environment meets the following requirements:

- The configuration of the app **Nextcloud** is complete and done.
- The SAML sign-in for **Nextcloud** works with **SimpleSAMLphp** as SAML IDP[69].
- The UCS domain has the latest version of the app **Keycloak** installed.

To setup **Nextcloud** for SAML with **Keycloak**, use the following steps:

1. To create a SAML client for **Nextcloud**, run the following commands on the UCS system that has the **Keycloak** app installed. Replace *NEXTCLOUD_FQDN* with the FQDN (fully qualified domain name) of the UCS system that has **Nextcloud** installed.

Listing 4.1: Create SAML client for *Nextcloud* in *Keycloak*

```
$ NEXTCLOUD_FQDN="REPLACE with NEXTCLOUD_FQDN"
$ univention-keycloak saml/sp create \
  --metadata-url="https://$NEXTCLOUD_FQDN/nextcloud/apps/user_saml/saml/
↪metadata" \
  --role-mapping-single-value
```

2. To obtain the **Keycloak** base URL and certificate, run the following commands on the UCS system that has **Keycloak** installed:

[66] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[67] https://docs.software-univention.de/keycloak-app/latest/configuration.html#login-portal
[68] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[69] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP

Listing 4.2: Retrieve public certificate of *Keycloak*

```
$ univention-keycloak saml/idp/cert get \
   --as-pem \
   --output "/tmp/keycloak.cert"
```

The output of the command in Listing 4.2 saves the certificate in the file `/tmp/keycloak.cert`. Copy this file to the UCS system that has the **Nextcloud** app installed.

Listing 4.3: Retrieve public certificate and *Keycloak* base URL

```
$ univention-keycloak get-keycloak-base-url
```

The command in Listing 4.3 outputs the base URL of your **Keycloak** server. Replace *SSO_URL* in the following instruction with this value.

3. To change the IdP settings for **Nextcloud**, run the following commands on the UCS system that has it installed. Copy the certificate file `/tmp/keycloak.cert` from Listing 4.2 to the UCS system with **Nextcloud** installed and replace *SSO_URL*.

Listing 4.4: Configure the **Nextcloud** app to use **Keycloak** as IdP

```
$ SSO_URL="REPLACE WITH SSO_URL"
$ univention-app shell nextcloud sudo -u www-data /var/www/html/occ␣
↪saml:config:set \
  --idp-x509cert="$(cat /tmp/keycloak.cert)" \
  --general-uid_mapping="uid" \
  --idp-singleLogoutService.url="$SSO_URL/realms/ucs/protocol/saml" \
  --idp-singleSignOnService.url="$SSO_URL/realms/ucs/protocol/saml" \
  --idp-entityId="$SSO_URL/realms/ucs" 1
```

To validate the setup, visit the sign-in page of your **Nextcloud** app and initiate a single sign-on. **Nextcloud** redirects you to **Keycloak** for authentication. You can use **Nextcloud** after authentication.

See also:

Nextcloud[70]
in Univention App Catalog

4.1.3 Google Workspace Connector

This section provides a step-by-step guide for migrating the **Google Connector** app to use **Keycloak** as SAML IDP[71]. The migration assumes that your environment meets the following requirements:

- The configuration of the app **Google Workspace Connector** is complete and done.
- The SAML login for your *Google Workspace* works with **SimpleSAMLphp**.
- You have the administrator credentials for your *Google Workspace Admin Console* for the SAML service configuration.
- The UCS domain has the latest version of the app **Keycloak** installed.

To setup **Google Workspace Connector** for SAML with **Keycloak** use the following steps:

1. Create a SAML SP[72], the client, for **Google Workspace Connector** in **Keycloak**. Run the following commands on the UCS system that has **Keycloak** installed:

[70] https://www.univention.com/products/univention-app-center/app-catalog/nextcloud/
[71] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[72] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

Listing 4.5: Create SAML client for *Google Workspace Connector* in *Keycloak*

```
$ google_domain="REPLACE_WITH_NAME_OF_YOUR_GOOGLE_DOMAIN"
$ univention-keycloak saml/sp create \
    --client-id google.com \
    --assertion-consumer-url-post "https://www.google.com/a/$google_domain/acs
↪" \
    --single-logout-service-url-post "https://www.google.com/a/$google_domain/
↪acs" \
    --idp-initiated-sso-url-name google.com \
    --name-id-format email \
    --frontchannel-logout-off
$ univention-keycloak user-attribute-ldap-mapper \
  create univentionGoogleAppsPrimaryEmail
$ univention-keycloak saml-client-nameid-mapper create \
  google.com \
  univentionGoogleAppsPrimaryEmail \
  --user-attribute univentionGoogleAppsPrimaryEmail \
  --mapper-nameid-format urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress
```

2. For the configuration of the SAML settings of your *Google Workspace* you need the public certificate and the base URL of the **Keycloak** server. Run the following commands on the UCS system that has **Keycloak** installed:

Listing 4.6: Retrieve public certificate and *Keycloak* base URL

```
$ univention-keycloak saml/idp/cert get --as-pem --output /tmp/keycloak.cert
```

The output of the command in Listing 4.6 is the certificate. Copy the file `/tmp/keycloak.cert` to your local client computer.

Listing 4.7: Retrieve public certificate and *Keycloak* base URL

```
$ univention-keycloak get-keycloak-base-url
```

The command in Listing 4.7 outputs the base URL of your **Keycloak** server. Replace *SSO_URL* in the following instruction with this value.

3. Change the *Third-party SSO profile for your organisation* settings in the *Google Workspace Admin console* of your google domain.
    1. Open the URL https://admin.google.com and login with your administrator account.
    2. Go to *Security* ‣ *Authentication* ‣ *SSO with third-party IdP*.
    3. Open *Third-party SSO profile for your organisation*.
    4. Change *Sign-in page URL* to `SSO_URL/realms/ucs/protocol/saml`.
    5. Change *Sign-out page URL* to `SSO_URL/realms/ucs/protocol/openid-connect/logout`.
    6. To upload the **Keycloak** certificate in `/tmp/kc.cert` from Listing 4.6, click *REPLACE CERTIFICATE*.
    7. To activate the settings, click *Save*.
4. Change the link in the UCS portal entry *Google Workspace login* for the IdP initiated single sign-on. On your UCS *Primary Directory Node* run the following commands:

Listing 4.8: Change portal entry for *Google Workspace login* to IdP initiated single sign-on

```
$ google_domain="REPLACE WITH NAME_OF_YOUR_GOOGLE_DOMAIN"
$ SSO_URL="REPLACE WITH SSO_URL"
$ udm portals/entry modify \
  --dn "cn=SP,cn=entry,cn=portals,cn=univention,$(ucr get ldap/base)" \
  --set link='"en_US" "'"$SSO_URL"'/realms/ucs/protocol/saml/clients/google.
↪com?RelayState=https://www.google.com/a/'"$google_domain"'/ServiceLogin"'
```

To validate the setup, visit https://google.com and sign in with one of the UCS user accounts enabled for *Google Workspace*. Also, verify the UCS portal entry *Google Workspace login* for the IdP initiated single sign-on.

Warning: The automatic redirect after the single sign-out doesn't work with **Keycloak**.

See also:

Google Workspace Connector[Page 20, 73]
in Univention App Catalog

4.1.4 Microsoft 365 Connector

The example illustrates how to migrate the app **Microsoft 365 Connector** to use **Keycloak** as SAML IDP[74]. It assumes that your environment meets the following requirements:

- The configuration of the app **Microsoft 365 Connector** is complete and done.
- The SAML login for your *Azure Active Directory* domain works with **SimpleSAMLphp**.
- You have a client computer with *Microsoft Windows* installed on it and a working internet connection on the client computer to configure SAML in *Azure Active Directory*.
- You have the *Administrator* credentials for your *Azure Active Directory* domain for the SAML service configuration.
- The UCS domain has the latest version of the app **Keycloak** installed.

To setup **Microsoft 365 Connector** for SAML with **Keycloak**, use the following steps:

1. Create a SAML SP[75], the client, for **Microsoft 365 Connector** in **Keycloak**. Run the following commands on the UCS system that has **Keycloak** installed:

Listing 4.9: Create SAML client for *Microsoft 365 Connector* in *Keycloak*

```
# get the saml client metadata xml from microsoft
$ curl https://nexus.microsoftonline-p.com/federationmetadata/saml20/
↪federationmetadata.xml > /tmp/ms.xml

# create the client in keycloak
$ univention-keycloak saml/sp create \
  --metadata-file /tmp/ms.xml \
  --metadata-url urn:federation:MicrosoftOnline \
  --idp-initiated-sso-url-name MicrosoftOnline \
  --name-id-format persistent

# create a SAML nameid mapper
$ univention-keycloak saml-client-nameid-mapper create \
```

(continues on next page)

[73] https://www.univention.com/products/univention-app-center/app-catalog/google-apps/
[74] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-IDP
[75] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

(continued from previous page)

```
  urn:federation:MicrosoftOnline \
  entryUUID \
  --mapper-nameid-format "urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
↪" \
  --user-attribute entryUUID \
  --base64
```

2. For the configuration of the SAML settings of your *Azure Active Directory* domain you need the public certificate and the base URL of the **Keycloak** server. Run the following commands on the UCS system that has **Keycloak** installed:

Listing 4.10: Retrieve public certificate and *Keycloak* base URL

```
$ univention-keycloak saml/idp/cert get --output /tmp/keycloak.cert
$ cat /tmp/keycloak.cert
```

The output of the command in Listing 4.10 is the certificate. Replace *KEYCLOAK_CERTIFICATE* in the following steps with this value.

Listing 4.11: Retrieve public certificate and *Keycloak* base URL

```
$ univention-keycloak get-keycloak-base-url
```

The command in Listing 4.11 outputs the base URL of your **Keycloak** server. Replace *SSO_URL* in the following steps with this value.

3. To configure the SAML settings for the *Azure Active Directory* domain, copy the following code block to your *Microsoft Windows* client computer. Replace the values for *SSO_URL*, *KEYCLOAK_CERTIFICATE*, the *Azure Active Directory* domain name and credentials and run the script in the *Microsoft Windows* **PowerShell**.

Listing 4.12: Change *Azure Active Directory* domain authentication to use *Keycloak*

```
# CHANGE this according to your setup/environemt
$sso_url = "replace with SSO_URL"
$signing_cert = "replace with KEYCLOAK_CERTIFICATE"
$domain = "YOUR AZURE DOMAIN NAME"
$username = "YOUR AZURE DOMAIN ADMIN"
$password = "PASSWORD OF YOUR AZURE DOMAIN ADMIN"
$realm = "REALM OF CHOICE (usually ucs)"
# CHANGE end

$issuer_uri = "$sso_url/realms/$realm"
$logon_uri = "$sso_url/$realm/ucs/protocol/saml"
$passive_logon_uri = "$sso_url/realms/$realm/protocol/saml"
$logoff_uri = "$sso_url/realms/$realm/protocol/saml"
$pass = ConvertTo-SecureString -String "$password" -AsPlainText -Force
$credential = New-Object -TypeName System.Management.Automation.PSCredential -
↪ArgumentList $username, $pass
$o365cred = Get-Credential $credential

Install-Module MSOnline
Import-Module MSOnline
Connect-MsolService -Credential $o365cred

Set-MsolDomainAuthentication -DomainName "$domain" -Authentication Managed
Set-MsolDomainAuthentication `
    -DomainName "$domain" `
    -FederationBrandName "UCS" `
    -Authentication Federated `
```

(continues on next page)

(continued from previous page)

```
    -ActiveLogOnUri "$logon_uri" `
    -PassiveLogOnUri "$passive_logon_uri" `
    -SigningCertificate "$signing_cert" `
    -IssuerUri "$issuer_uri" `
    -LogOffUri "$logoff_uri" `
    -PreferredAuthenticationProtocol SAMLP

Get-MsolDomain
Pause
```

4. To change the link in the UCS portal entry *Microsoft 365 Login* for the IdP initiated single sign-on, run the following commands on your UCS *Primary Directory Node*:

Listing 4.13: Change portal entry for *Microsoft 365 Login* to IdP initiated single sign-on

```
$ SSO_URL="REPLACE WITH SSO_URL"
$ udm portals/entry modify \
  --dn "cn=office365,cn=entry,cn=portals,cn=univention,$(ucr get ldap/base)" \
  --set link='"en_US" "'"$SSO_URL"'/realms/$realm/protocol/saml/clients/
↪MicrosoftOnline"'
```

To validate the setup, visit https://www.microsoft365.com/ and sign in with one of the UCS user accounts enabled for *Microsoft 365*. Also, verify the UCS portal entry *Microsoft 365 Login* for the IdP initiated single sign-on.

Warning: The automatic redirect after the single sign-out doesn't work with **Keycloak**.

Migration of additional Azure AD connections

It is possible to configure additional Azure AD connections for single-sign on using the Microsoft 365 Connector wizard. If multiple AD connections were configured according to Extended Configuration[76], each connection needs to be migrated individually to **Keycloak**. Since Azure AD explicitly needs different entity IDs for each connection, this entails the creation of a new IdP and therefore a new realm for each connection.

To create a new logical IdP in **Keycloak**, run the following commands on your UCS **Keycloak** host.

Listing 4.14: Create a new logical IdP in Keycloak

```
$ AD_CONNECTION="REPLACE WITH MICROSOFT 365 AD CONNECTION NAME"
$ univention-keycloak proxy-realms create "$AD_CONNECTION"
```

Use the following call to get the certificate of your newly created IdP

Listing 4.15: Get the certificate of the newly created Realm

```
$ univention-keycloak saml/idp/cert get \
   --realm-id="$AD_CONNECTION" \
   --output "/tmp/keycloak-"$AD_CONNECTION".cert"
 cat /tmp/keycloak-"$AD_CONNECTION".cert
```

Using this certificate as `$signing_cert` and the `AD_CONNECTION` as `$realm`, follow the steps from *Change Azure Active Directory domain authentication to use Keycloak* (page 21) onward to update the SAML settings for the *Azure Active Directory* domain.

Repeat these steps for each additional configured Azure AD connection.

See also:

[76] https://docs.software-univention.de/manual/5.0/en/domain-ldap/saml.html#domain-saml-extended-configuration

Microsoft 365 Connector Page 23, 77
in Univention App Catalog

4.2 Services using OIDC

The following examples demonstrate the migration of services that use OIDC for authentication and **OpenID Connect Provider** as IdP to **Keycloak** as IdP.

4.2.1 ownCloud

This section is about the migration of the **ownCloud** app to use **Keycloak** as OIDC Provider[78] for authentication. It assumes that your environment meets the following requirements:

- The configuration of the app **ownCloud** is complete and done.
- The OIDC sign-in for **ownCloud** works with **OpenID Connect Provider** as OIDC Provider[79].
- The UCS domain has the latest version of the app **Keycloak** installed.

To setup **ownCloud** for OIDC with **Keycloak** use the following steps:

1. To obtain the necessary information such as `clientsecret` and `redirectURI`, run the following command on the UCS *Primary Directory Node*. You need the values to create the OIDC RP[80], the client, in the next step.

   Listing 4.16: Get current settings for the *ownCloud* OIDC client

   ```
   $ udm oidc/rpservice list --filter name=owncloud
     DN: cn=owncloud,cn=oidc,cn=univention,dc=...
       applicationtype: web
       clientid: owncloud
       clientsecret: -> copy this value
       insecure: None
       name: owncloud
       redirectURI: -> copy this value
       trusted: yes
   ```

   Look for the values of `clientsecret` and `redirectURI` and copy them, for example, into a temporary text file.

2. To create the OIDC RP[81], the client, for **ownCloud** in **Keycloak**, run the following commands on the UCS system that has **Keycloak** installed. Replace *clientsecret* and *redirectURI* with the values for these settings in Listing 4.16 from the previous step.

   Listing 4.17: Create OIDC client for *ownCloud* in *Keycloak*

   ```
   $ CLIENT_SECRET="REPLACE WITH clientsecret"
   $ REDIRECT_URI="REPLACE WITH redirectURI"
   $ univention-keycloak oidc/rp create \
       --client-secret "$CLIENT_SECRET" \
       --app-url "$REDIRECT_URI" owncloud
   ```

3. To obtain the base URL of your **Keycloak** server, run the following command on the UCS system that has it installed:

[77] https://www.univention.com/products/univention-app-center/app-catalog/microsoft365/
[78] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[79] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-Provider
[80] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-RP
[81] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-RP

Listing 4.18: Obtain *Keycloak* base URL

```
$ univention-keycloak get-keycloak-base-url
```

Replace *SSO_URL* in the following step with this value.

4. Change the IDP setting in **ownCloud**. Run the following command on the UCS system that has **ownCloud** installed:

Listing 4.19: Change IDP settings in *ownCloud*

```
$ SSO_URL="REPLACE WITH SSO_URL"
$ univention-app configure owncloud \
  --set OWNCLOUD_OPENID_PROVIDER_URL="$SSO_URL/realms/ucs"
```

To validate the setup, visit the sign-in page of your **ownCloud** app and initiate a single sign-on. **ownCloud** redirects you to **Keycloak** for authentication. You can use **ownCloud** after authentication.

See also:

ownCloudPage 24, 82
in Univention App Catalog

[82] https://www.univention.com/products/univention-app-center/app-catalog/owncloud/

CHAPTER

FIVE

SERVICES VALIDATION AND TROUBLESHOOTING

Every procedure during the migration has the potential to fail at some point. This section provides hints for troubleshooting such situations.

5.1 Log files

As first step, review the log files of **Keycloak** and the connected services:

- Review the log files of **Keycloak** and look for errors. On the UCS system that has **Keycloak** installed, run the following command:

```
$ univention-app logs keycloak
```

- For an extensive troubleshooting guide of the **Keycloak** app, refer to Troubleshooting[83] in the *Univention Keycloak app documentation* [1].
- Review the log files of the following services:
  - For UMC: `/var/log/univention/management-console-web-server.log`
  - For **Nextcloud**: `/var/lib/univention-appcenter/apps/nextcloud/data/nextcloud-data/nextcloud.log`
  - For **ownCloud**: `/var/lib/univention-appcenter/apps/owncloud/data/files/owncloud.log`
  - For other services: consult the manual of the service

5.2 Single sign-on settings

Also verify the configuration of **Keycloak** and the single sign-on settings of the services:

- For OIDC services make sure that the service has the correct `clientsecret` and `clientid`. The values for these settings must match in **Keycloak** and the services.
- For SAML services verify that the service uses the current, public certificate of the **Keycloak** server.
- For SAML verify that the `clientid` in the **Keycloak** configuration of your SAML SP[84] is correct. This is also the issuer for the SAML authentication request. The value is service specific, but needs to match the expectations of the service.

Additionally, verify the following items:

- Ensure that all involved systems have the same and synchronized time.
- Use *Developer Tools* of your browser to see which requests fail to narrow down the cause of the problem.

[83] https://docs.software-univention.de/keycloak-app/latest/troubleshooting.html#app-troubleshooting
[84] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-SAML-SP

CHAPTER

SIX

PREPARE FOR THE UPDATE TO UCS 5.2

This section explains the necessary steps to prepare your domain for the update to UCS 5.2. Follow these steps only after

- you migrated all services that use single sign-on for authentication to use **Keycloak** as IdP,
- or you are absolutely sure no service uses **SimpleSAMLphp** or **OpenID Connect Provider** as IdP for single sign-on authentication.

UCS before version 5.2 stored all the **SimpleSAMLphp** and **OpenID Connect Provider** client configurations as UDM modules `saml/serviceprovider` and `oidc/rpservice`. The update to 5.2 blocks until all these objects are removed.

Warning: After removing these client objects, single sign-on with **SimpleSAMLphp** or **OpenID Connect Provider** does no longer work.

To prepare your domain for the update to UCS 5.2, run the following command on your UCS Primary Directory Node[85] to backup all IdP client object settings and subsequently remove them:

Listing 6.1: Remove single sign-on client objects from UDM

```
$ univention-keycloak-migration-status --delete
```

[85] https://docs.software-univention.de/manual/5.0/en/domain-ldap/system-roles.html#domain-ldap-primary-directory-node

CHAPTER

SEVEN

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] *Univention Keycloak app documentation*. Univention GmbH, 2023. URL: https://docs.software-univention.de/keycloak-app/latest/.

[2] *UCS 5.0 Manual*. Univention GmbH, 2021. URL: https://docs.software-univention.de/manual/5.0/en/.

[3] Raphaël Hertzog and Roland Mas. *The Debian Administrator's Handbook*, chapter Shell and Basic Commands. Freexian SARL, First edition, 2020. URL: https://www.debian.org/doc/manuals/debian-handbook/short-remedial-course.en.html#sect.shell-and-basic-commands.

[4] *Managing OpenID Connect and SAML Clients*. URL: https://www.keycloak.org/docs/23.0.7/server_admin/#assembly-managing-clients_server_administration_guide.

INDEX

E

K

S

U