

OX Connector for Nubus documentation 4.x

Release 4.x

Univention GmbH

Sep 29, 2026

The source of this document is licensed under [GNU Affero General Public License v3.0](#) only.

CONTENTS

1	OX Connector for Nubus	1
1.1	Deployment paths	1
1.2	Audience and prerequisites	1
1.3	Document scope	1
1.4	Support status	2
1.5	Feedback	2
2	Prerequisites	3
2.1	OX App Suite server	3
2.2	OX Connector app in Univention App Center	4
2.3	Nubus for Kubernetes	4
3	Installation	5
3.1	Install on Nubus for UCS	5
3.1.1	Install with a web browser	5
3.1.2	Install from the command line	6
3.2	Install on Nubus for Kubernetes	7
3.2.1	Prepare OX App Suite	7
3.2.2	Install the packaged integration	7
3.2.3	Create a Provisioning API subscription	9
3.2.4	Prepare the OX Consumer configuration	11
3.2.5	Install the OX Consumer	14
4	Configuration reference	17
4.1	Configuration for UCS	17
4.1.1	App settings	17
4.1.2	UCR variables	21
4.1.3	User attribute mapping	22
4.2	Import additional CA certificates on UCS	25
4.3	Structured logging on UCS	25
4.4	Configuration for Nubus for Kubernetes	26
4.4.1	Configure a database	26
4.4.2	Helm Chart references values	26
4.4.3	affinity	27
4.4.4	environment	27
4.4.5	extraVolumeMounts	27
4.4.6	extraVolumes	27
4.4.7	fullnameOverride	27
4.4.8	global	27
4.4.9	nameOverride	28
4.4.10	nodeSelector	28
4.4.11	openXchange	28
4.4.12	oxConnector	29
4.4.13	persistence	30

4.4.14	podAnnotations	30
4.4.15	podSecurityContext	30
4.4.16	probes	30
4.4.17	provisioningApi	31
4.4.18	replicaCount	32
4.4.19	resources	32
4.4.20	resourcesWaitForDependency	32
4.4.21	securityContext	32
4.4.22	serviceAccount	33
4.4.23	tolerations	33
4.4.24	waitForDependency	33
5	Usage	35
5.1	Manage OX objects	35
5.1.1	Contexts	35
5.1.2	Users	35
5.1.3	Groups	36
5.1.4	Access profiles	37
5.1.5	Functional accounts	37
5.1.6	Resources	38
5.2	Shared accounts	38
5.2.1	Management module for shared accounts	38
5.2.2	Management module for permissions	39
5.2.3	Migration from functional accounts to shared accounts	39
6	Architecture	43
6.1	Shared architecture	43
6.1.1	Common architecture components	43
6.1.2	Provisioned attributes	44
6.1.3	Database of object state	44
6.2	For Nubus for UCS	44
6.2.1	Overview	44
6.2.2	How the connector works	45
6.2.3	Provisioned attributes	47
6.2.4	Database of stored object state	48
6.3	For Nubus for Kubernetes	48
6.3.1	Overview	48
6.3.2	Deployment	48
6.3.3	OX Connector Provisioning Consumer behavior	49
6.3.4	Provisioned attributes	52
6.3.5	Database of stored object state	52
7	Limitations	53
7.1	OX Connector integration with the OX App Suite app	53
7.2	How OX Connector handles faulty items	53
7.2.1	OX Connector continues after faulty items	53
7.2.2	OX Connector stops at faulty items	54
7.3	No validation of access profile rights	54
8	Troubleshooting	55
8.1	Manage provisioning tasks	55
8.1.1	Check provisioning health	56
8.1.2	Handle failed tasks	57
8.1.3	Resolve blocked provisioning	59
8.1.4	Re-provision all data	60
8.2	Troubleshoot OX Connector on Nubus for UCS	63
8.2.1	Log files	63
8.2.2	Ensure OX database ID integrity	64
8.2.3	Duplicate display names	65

8.2.4	Missing group members	66
8.2.5	Invalid values for OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER	66
8.3	Troubleshoot OX Connector on Nubus for Kubernetes	68
8.3.1	Log files	68
8.3.2	Ensure OX database ID integrity	69
8.3.3	Missing group members	70
8.3.4	Invalid values for OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER	71
8.4	Troubleshoot migration from functional accounts to shared accounts	73
8.5	Collect information for a support ticket	74
9	Changelog	75
9.1	Changelog OX Connector app	75
9.1.1	v4.0.0	75
9.1.2	v3.2.3	76
9.1.3	v3.2.2	76
9.1.4	v3.2.1	76
9.1.5	v3.2.0	76
9.1.6	v3.1.0	76
9.1.7	v3.0.2	77
9.1.8	v3.0.1	77
9.1.9	v3.0.0	77
9.1.10	v2.3.5	77
9.1.11	v2.3.4	78
9.1.12	v2.3.3	78
9.1.13	v2.3.2	78
9.1.14	v2.3.1	78
9.1.15	v2.3.0	79
9.1.16	v2.2.15	79
9.1.17	v2.2.14	79
9.1.18	v2.2.13	80
9.1.19	v2.2.12	80
9.1.20	v2.2.11	80
9.1.21	v2.2.10	80
9.1.22	v2.2.9	80
9.1.23	v2.2.8	80
9.1.24	v2.2.7	81
9.1.25	v2.2.6	81
9.1.26	v2.2.5	81
9.1.27	v2.2.4	81
9.1.28	v2.2.3	81
9.1.29	v2.2.2	81
9.1.30	v2.2.1	82
9.1.31	v2.2.0	82
9.1.32	v2.1.4	82
9.1.33	v2.1.3	82
9.1.34	v2.1.2	82
9.1.35	v2.1.1	83
9.1.36	v2.1.0	83
9.1.37	v2.0.1	83
9.1.38	v2.0.0	83
9.1.39	v1.1.0	83
9.2	Changelog OX Connector on Kubernetes	84
9.2.1	v0.41.1	84
9.2.2	v0.41.0	84
9.2.3	v0.36.2	85
9.2.4	v0.36.1	86
9.2.5	v0.36.0	86

10 Document changelog	87
10.1 Year 2026	88
11 Bibliography	89
Bibliography	91
Index	93

OX CONNECTOR FOR NUBUS

The *OX Connector* sends selected directory objects from Nubus to a remote *OX App Suite* installation through the OX SOAP API. These objects include user accounts, user groups, and resources.

Nubus is the leading system for identity data. When you create, change, or remove selected objects in Nubus, the *OX Connector* synchronizes these changes to *OX App Suite*.

The connector doesn't install or operate *OX App Suite*.

1.1 Deployment paths

The *OX Connector* has two deployment paths:

Connector app in Univention App Center

Use this deployment path if you operate Nubus on Univention Corporate Server (UCS).

The *OX Connector app* runs as an App Center app on UCS. It sends selected UCS directory objects to *OX App Suite*.

Nubus for Kubernetes

Use this deployment path if you operate Nubus for Kubernetes.

The *OX Consumer* runs in your Kubernetes cluster. It sends selected Nubus directory objects to *OX App Suite*. The packaged integration for *OX App Suite* extends Nubus with the required directory schema and management interface customizations.

Both deployment paths use the same provisioning logic. The chapters in this manual state which steps apply to each deployment path.

1.2 Audience and prerequisites

This manual is for technical administrators and operators who want to connect Nubus with *OX App Suite*. You need administrative access to the Nubus deployment that you want to connect. You also need access to an existing *OX App Suite* installation, including the credentials and network access that the connector needs to reach the OX SOAP API.

For the App Center deployment, you need to know how to use a shell on Nubus for UCS. You also need to know how to manage apps through the Univention App Center.

For the Kubernetes deployment, you need to know Kubernetes concepts. You also need to know how to deploy and manage applications with Helm.

1.3 Document scope

This manual covers the following topics:

- Install the *OX Connector* for the supported deployment paths.
- Configure provisioning from Nubus to *OX App Suite*.
- Understand how the connector works.

- Read about known limitations.
- Find log files and troubleshoot common problems.

This manual doesn't cover installation, setup, or operation of the following products:

- Kubernetes
- *OX App Suite*
- *Nubus for UCS*
- *Nubus for Kubernetes*

See also

Nubus for UCS

For information about installation, setup, and operation of UCS, see [Nubus for UCS Operation Manual¹](#) in *Nubus for UCS 5.2 - Operation Manual* [1].

Nubus for Kubernetes

For information about installation, setup, and operation of Nubus for Kubernetes, see [Deploy Nubus for Kubernetes²](#) in *Nubus for Kubernetes - Operation Manual 1.x* [2].

OX App Suite

For information about setup and operation of *OX App Suite*, see *App Suite Admin Guide 7.10.6* [3].

1.4 Support status

Univention supports the *OX Connector* app for Nubus for UCS and the *OX Connector* for Nubus for Kubernetes. Support for the connector doesn't include installation, configuration, or operation of *OX App Suite*.

For more information about Nubus for Kubernetes, see [Univention Nubus³](#).

1.5 Feedback

We welcome your feedback about this documentation. If you have comments, suggestions, or criticism, [submit your feedback⁴](#) to help improve this document.

¹ <https://docs.software-univention.de/ucs-operation/5.2/en/lifecycle/index.html#lifecycle>

² <https://docs.software-univention.de/nubus-kubernetes-operation/1.x/en/deploy.html#nubus-deployment-all-deps>

³ <https://docs.software-univention.de/n/en/nubus.html#page-nubus>

⁴ <https://www.univention.com/feedback/?ox-connector=generic>

PREREQUISITES

Before you install the *OX Connector*, make sure that your Nubus deployment and *OX App Suite* meet the prerequisites.

2.1 OX App Suite server

The *OX App Suite* server must meet the following prerequisites:

1. Installed OX App Suite instance

You need an existing *OX App Suite* instance. The *OX Connector* doesn't install, configure, or operate *OX App Suite*.

For information about installing *OX App Suite*, see *App Suite Admin Guide 7.10.6* [3].

2. SOAP API access

The *OX App Suite* instance must allow SOAP requests so that the *OX Connector* can access the `/webservices` endpoint in *OX App Suite*.

1. OX App Suite administrator

Set up an administrator user in *OX App Suite* who can create OX contexts. The *OX Connector* uses this user to manage OX contexts. In the connector configuration, enter the username and password for this user. For information about managing OX contexts manually, see *App Suite Admin Guide 7.10.6* [3].

For the *OX Connector* app in Univention App Center, use the settings `OX_MASTER_ADMIN` (page 18) and `OX_MASTER_PASSWORD` (page 18).

If *OX Connector* and *OX App Suite* run on separate UCS systems, retrieve the password from the `/etc/ox-secrets/master.secret` file on the UCS system that runs *OX App Suite*. Use `oxadminmaster` as the administrator username.

For Nubus for Kubernetes, configure the OX administrator credentials in the Helm values for the *OX Consumer*.

For manually managing OX contexts without the OX Connector, see *Contexts* (page 35).

2. Duplicate display names

Allow duplicate display names in *OX App Suite*. Add the lines in [Listing 2.1](#) to the `user.properties` file.

Listing 2.1: Allow duplicate display names in *OX App Suite*

```
com.openexchange.user.enforceUniqueDisplayName=false
com.openexchange.folderstorage.database.preferDisplayName=false
```

3. Group names

OX App Suite must allow all group names that administrators can use in Nubus. Add the line in [Listing 2.2](#) to the `Group.properties` file.

Listing 2.2: Allow all group names in *OX App Suite*

```
CHECK_GROUP_UID_FOR_NOT_ALLOWED_CHARS=false
```

2.2 OX Connector app in Univention App Center

The *OX Connector* app from Univention App Center needs the *referential integrity* overlay in the central LDAP directory.

The overlay keeps the Univention Directory Manager (UDM) objects that the *OX Connector* uses consistent. It also ensures that these UDM objects reference user objects correctly.

For details, see *Referential Integrity in OpenLDAP Software 2.4 Administrator's Guide* [4].

OX Connector on UCS Primary Directory Node

If you install the *OX Connector* app from Univention App Center on [UCS Primary Directory Node](#)⁵, the app activates the *referential integrity* overlay. You don't need to take further action.

OX Connector on other system roles

If you install the *OX Connector* app from Univention App Center on a Nubus for UCS system role other than the [UCS Primary Directory Node](#)⁶, run the commands in [Listing 2.3](#) on the [UCS Primary Directory Node](#)⁷ with root privileges.

Listing 2.3: Activate the OpenLDAP *referential integrity* overlay on the UCS Primary Directory Node

```
$ ucr set ldap/refint=true
$ service slapd restart
```

2.3 Nubus for Kubernetes

On Nubus for Kubernetes, the *OX Connector* runs as the *OX Consumer*.

Before you install the *OX Consumer*, install the packaged integration for *OX App Suite*. Follow [Load packaged integrations](#)⁸. The packaged integration installs the required LDAP schema in the *Directory Service*. It also adds the required customizations to the *Management UI* in Nubus for user accounts, user groups, and resources.

Before you install the packaged integration, ask the team that provides it for the container image details.

You need the registry name and the repository name. This documentation uses the following example values:

Registry

```
artifacts.software-univention.de
```

Repository

```
nubus/images/ox-extension
```

For details, see *Univention Nubus - Customization and Modification Manual* [5].

⁵ <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-UCS-Primary-Directory-Node>

⁶ <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-UCS-Primary-Directory-Node>

⁷ <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-UCS-Primary-Directory-Node>

⁸ <https://docs.software-univention.de/nubus-customization/1.x/en/packaged-integrations/load.html#nubus-packaged-integrations-load>

INSTALLATION

Before you install the *OX Connector*, make sure that your environment meets the prerequisites. For details, see *Prerequisites* (page 3).

3.1 Install on Nubus for UCS

As an administrator, you can install the *OX Connector* app with Univention App Center.

Before you start, make sure that your environment meets the *OX Connector app in Univention App Center* (page 4). Identify the required app settings.

You can install the app in Nubus for UCS in the following ways:

- Install with a web browser through the *Management UI*. See *Install with a web browser* (page 5).
- Install from the command line on Nubus for UCS. See *Install from the command line* (page 6).

See also

Univention App Center⁹

in *Nubus for UCS 5.2 - Operation Manual* [1] to learn how to install apps with Univention App Center.

3.1.1 Install with a web browser

To install the *OX Connector* with the *Management UI* in Nubus, use the following steps.

Before you start, make sure that you have a user account with access to administrative management modules.

For more information, see *Choose the right user account*¹⁰ in *Nubus Manual 1.x* [6].

1. In a web browser, [sign in](#)¹¹ to the *Management UI*.
2. Open the *App Center* management module.
3. Select *OX Connector* or search for it, and then open the app.
4. Click *Install*.
5. Configure the *App settings* for your environment. For a reference, see *App settings* (page 17).
6. Click *Start Installation*.
7. Verify that App Center shows the *OX Connector* app as installed.

See also

⁹ <https://docs.software-univention.de/ucs-operation/5.2/en/lifecycle/app-center.html#lifecycle-app-center>

¹⁰ <https://docs.software-univention.de/nubus-manual/1.x/en/auth.html#nubus-authentication-sign-in-choose-user-account>

¹¹ <https://docs.software-univention.de/nubus-manual/1.x/en/auth.html#nubus-authentication-sign-in>

How to install applications¹²

in *Nubus for UCS 5.2 - Operation Manual* [1] for information about app installations.

3.1.2 Install from the command line

To install the *OX Connector* app from the command line, use the following steps:

1. Open a terminal or connect to a remote shell. Use a user account that has administrative rights, for example `root`.
2. Determine the app settings for your environment. To pass custom settings to the app during installation, use the command template in [Listing 3.1](#).

Replace the following placeholders:

- `SETTING_KEY` with the app setting name.
- `SETTING_VALUE` with the value.

For an example, see [Listing 3.2](#).

For a reference, see [App settings](#) (page 17).

3. Prepare credentials for the installation command. Before you run the installation command, make sure that you can provide the password for the domain administrator account, such as `Administrator`.

Tip

To use a different account, pass the account name with `--username` and the password file with `--pwd-file`. For more information, see `univention-app install -h`.

4. Install the *OX Connector* app with the `univention-app install` command.

The following command shows the syntax for passing app settings during installation.

Listing 3.1: Install *OX Connector* from the command line

```
$ univention-app install ox-connector --set SETTING_KEY=SETTING_VALUE
```

The following example shows multiple app settings in one installation command.

Listing 3.2: Example: Set app settings for *OX Connector* during installation

```
$ univention-app install ox-connector --set \
  OX_MASTER_ADMIN="oxadminmaster" \
  OX_MASTER_PASSWORD="SECURE_PASSWORD" \
  LOCAL_TIMEZONE="Europe/Berlin" \
  OX_LANGUAGE="de_DE" \
  DEFAULT_CONTEXT="10" \
  OX_SMTP_SERVER="smtp://my-smtp.example.com:587" \
  OX_IMAP_SERVER="imap://my-imap.example.com:143" \
  OX_SOAP_SERVER="https://my-ox.example.com"
```

Caution

Command-line passwords can appear in the shell history or process list. Use this method only on a trusted system.

¹² <https://docs.software-univention.de/ucs-operation/5.2/en/lifecycle/app-center.html#lifecycle-app-center-installation>

5. Verify that App Center shows the *OX Connector* app as installed. You must check the installed state in App Center.

3.2 Install on Nubus for Kubernetes

Provisioning sends selected directory objects from the *Directory Service* in Nubus to a remote *OX App Suite* installation. These objects include user accounts, user groups, and resources.

The *OX Consumer* sends this data through the *OX SOAP API*. For example, users created in Nubus appear in the *OX App Suite* address book. Users can also book synchronized resources, such as meeting rooms, in *OX App Suite*.

For Nubus for Kubernetes, the *OX Consumer* provides the same functionality for *OX App Suite* as the *OX Connector* app provides for the Nubus for UCS appliance. Both use the same business logic.

Before you start, make sure that your environment meets the Kubernetes prerequisites. For details, see *Nubus for Kubernetes* (page 4).

This page covers the following tasks:

1. *Prepare OX App Suite* (page 7)
2. *Install the packaged integration* (page 7)
3. *Create a Provisioning API subscription* (page 9)
4. *Prepare the OX Consumer configuration* (page 11)
5. *Install the OX Consumer* (page 14)

3.2.1 Prepare OX App Suite

If you don't have an *OX App Suite* installation, install it before you continue. Installing *OX App Suite* is beyond the scope of this manual. For installation resources, see the following links:

OX App Suite 8 Operations Guide¹³

for information about the deployment of OX App Suite 8.

OX App Suite 7¹⁴

for information about the installation of OX App Suite 7.

Important

Univention doesn't provide support for the installation of *OX App Suite*.

3.2.2 Install the packaged integration

This section shows how to install the packaged integration for *OX App Suite* in Nubus for Kubernetes. It adds these customizations to Nubus:

- A Nubus *Portal* tile for the *OX App Suite* instance.
- Management capabilities in the *Management UI* for the following *OX App Suite* objects:
 - Access profiles
 - Functional accounts
 - OX contexts
 - OX resources
 - User access to *OX App Suite*
 - User groups

¹³ <https://documentation.open-xchange.com/appsuite/operation-guides/>

¹⁴ https://wiki.open-xchange.com/wiki/index.php?title=AppSuite:Main_Page_AppSuite

For details, see [Load packaged integrations](#)¹⁵ in *Univention Nubus - Customization and Modification Manual* [5].

Before you continue, make sure that you know the container image registry and repository name for the packaged integration. For details, see *Nubus for Kubernetes* (page 4).

To install the packaged integration, follow these steps:

1. Copy the content from `nubus-values.yaml` to the `custom_values.yaml` file for your Nubus for Kubernetes installation. [Listing 3.3](#) shows the Helm Chart values to add.

oxDefaultContext

Define the number of the first and default context in *OX App Suite*. You can define the value as an integer, for example 10, or as a string, for example "10".

oxSystemUserPassword

Choose a secure password. Use the same password later when you set up LDAP in *Open-Xchange*.

portalOxLinkBase

Define the base URL of your *OX App Suite* instance. You get this URL after you install *OX App Suite*. See *Prepare OX App Suite* (page 7).

To choose an appropriate version number for the packaged integration in the `tag` attribute, see the [tags](#)¹⁶ in the repository.

Listing 3.3: Helm Chart values for adding the *Open-Xchange* packaged integration

```
# SPDX-License-Identifier: AGPL-3.0-only
# SPDX-FileCopyrightText: 2025-2026 Univention GmbH

global:
  extensions:
    - name: "ox"
      image:
        registry: "artifacts.software-univention.de"
        repository: "nubus/images/ox-extension"
        tag: "0.41.0"
        imagePullPolicy: "IfNotPresent"

nubusStackDataUms:
  templateContext:
    oxDefaultContext: "<The number of the first and default context in OX App Suite>"
    oxSystemUserPassword: "<Password for the LDAP search user>"
    portalOxLinkBase: "<Base URL to your OX App Suite instance>"
```

2. Apply the changes with Helm. Use the command in [Listing 3.4](#).

Before you run it, replace these placeholders:

- `<NAMESPACE>`: Kubernetes namespace of your Nubus installation.
- `<NUBUS_RELEASE_NAME>`: Helm Chart release name of your Nubus installation.
- `<NUBUS_VERSION>`: Nubus Helm Chart version to install.

Listing 3.4: Install the *Open-Xchange* packaged integration

```
$ export NAMESPACE_FOR_NUBUS="<NAMESPACE>"
$ export RELEASE_NAME="<NUBUS_RELEASE_NAME>"
$ export VERSION="<NUBUS_VERSION>"
```

(continues on next page)

¹⁵ <https://docs.software-univention.de/nubus-customization/1.x/en/packaged-integrations/load.html#nubus-packaged-integrations-load>

¹⁶ <https://github.com/univention/ox-connector/tags>

(continued from previous page)

```
$ helm upgrade \
  "$RELEASE_NAME" \
  --namespace="$NAMESPACE_FOR_NUBUS" \
  oci://artifacts.software-univention.de/nubus/charts/nubus \
  --values custom_values.yaml \
  --version "$VERSION"
```

3. Verify that the Helm upgrade completes successfully and that the packaged integration is available in Nubus for Kubernetes.

3.2.3 Create a Provisioning API subscription

Create the subscription before the *OX Consumer* connects to the *Provisioning Service*.

The subscription gives the *OX Consumer* access to the *Provisioning API*. The *Provisioning Service* sends directory object updates. It also provides the data that you want to provision.

For *OX App Suite*, the relevant directory objects are user accounts, user groups, and Open-Xchange resources. Examples include meeting rooms and functional mailboxes.

To create the *OX Consumer* subscription, follow these steps:

1. Make sure that your local client can reach the *Provisioning API*.

If you run the commands outside the cluster, create temporary port forwarding. Use the command in [Listing 3.5](#). Keep it running while you create the subscription.

Listing 3.5: Forward the *Provisioning API* to your local client

```
$ export LOCAL_PORT=7777
$ kubectl \
  --namespace "$NAMESPACE_FOR_NUBUS" \
  port-forward \
  services/"$RELEASE_NAME"-provisioning-api \
  "$LOCAL_PORT":80
```

2. Get the password for the *Provisioning API* administrator with the command in [Listing 3.6](#).

Caution

Environment variables can expose passwords in shell history, debug output, or process environments. Use this method only on a trusted system.

Listing 3.6: Retrieve the administrative password for the *Provisioning API*

```
$ export BASE_URL="http://localhost:7777"
$ export USERNAME="admin"
$ export PASSWORD="$(kubectl \
  --namespace "$NAMESPACE_FOR_NUBUS" \
  get secret \
  nubus-provisioning-api-admin \
  -o json \
  | jq -r ".data.password" \
  | base64 -d)"
```

3. Create the `provisioning-api.json` configuration file for the *OX Consumer*. Use the content in [Listing 3.7](#).

Set `password` to any value. Use the `name` and `password` values later for the *OX Consumer* configuration.

Listing 3.7: Subscription configuration for the *OX Consumer*

```
{
  "name": "ox-consumer",
  "realms_topics": [
    {
      "realm": "udm",
      "topic": "groups/group"
    },
    {
      "realm": "udm",
      "topic": "oxmail/accessprofile"
    },
    {
      "realm": "udm",
      "topic": "oxmail/functional_account"
    },
    {
      "realm": "udm",
      "topic": "oxmail/oxcontext"
    },
    {
      "realm": "udm",
      "topic": "oxresources/oxresources"
    },
    {
      "realm": "udm",
      "topic": "users/user"
    }
  ],
  "request_prefill": true,
  "password": "<your desired password>"
}
```

4. Send the subscription file to the *Provisioning API* with the command in Listing 3.8.

Listing 3.8: Create the subscription for the *OX Consumer*

```
$ curl \
  --user "$USERNAME":"$PASSWORD" \
  --request POST \
  "$BASE_URL"/v1/subscriptions \
  --header "Accept: application/json" \
  --header "Content-Type: application/json" \
  --data @provisioning-api.json
```

5. Verify that the subscription exists with the command in Listing 3.9.

Listing 3.9: Retrieve the list of subscriptions

```
$ curl \
  --user "$USERNAME":"$PASSWORD" \
  --request GET \
  "$BASE_URL"/v1/subscriptions \
  --header "Accept: application/json"
```

6. If you created a temporary port forward, stop the `kubect1 port-forward` command in Listing 3.5 after you verify that the subscription exists.

See also

Create subscription¹⁷

in *Univention Nubus - Customization and Modification Manual* [5] for information about how to create a subscription in the *Provisioning Service* using the *Provisioning API*.

3.2.4 Prepare the OX Consumer configuration

Prepare the configuration **before** you install the *OX Consumer* in a Kubernetes cluster. For the installation step, see *Install the OX Consumer* (page 14). The configuration defines the data source, the *Provisioning API*, and the data target, your *OX App Suite* instance.

To prepare the *OX Consumer* configuration, follow these steps:

1. Create the `ox-consumer-values.yaml` values file with the structure in Listing 3.10.

Listing 3.10: Configuration for the OX Consumer in the values file

```
# SPDX-License-Identifier: AGPL-3.0-only
# SPDX-FileCopyrightText: 2025 Univention GmbH

---
# Configuration for OX CONNECTOR
openXchange:
  # -- OX-Mail-Domain to generate OX-email-addresses
  domainName: null
  auth:
    # -- OX Admin username (the OX Admin can create, modify, delete contexts; ↪
    ↪has to exist)
    username: "oxadminmaster"
    # -- OX Admin password
    password: null
    existingSecret:
      # -- The name of an existing Secret to use for retrieving the password
      # for the ox admin password.
      #
      # "oxConnector.auth.password" will be ignored if this value is set.
      name: null
    keyMapping:
      # -- The key to retrieve the password from. Setting this value allows ↪
      ↪to use
      # a key with a different name.
      password: null
  # -- Default timezone for new users
```

(continues on next page)

¹⁷ <https://docs.software-univention.de/nubus-customization/1.x/en/api/provisioning.html#customization-api-provisioning-subscription>

(continued from previous page)

```

oxLocalTimezone: "Europe/Berlin"
# -- Default language for new users
oxLanguage: "de_DE"
# -- Default context for users (has to exist)
oxDefaultContext: "10"
# -- Default SMTP server for new users (if not set explicitly there)
oxSmtpServer: null
# -- Default IMAP server for new users (if not set explicitly there)
oxImapServer: null
# -- The server where Open-Xchange is installed
oxSoapServer: null
# -- OX Connector log level
# Chose from "DEBUG", "INFO", "WARNING" and "ERROR".
logLevel: "INFO"
# -- SQLAlchemy DB connection URL for the OX connector.
oxDbConnectionString: null

# Configuration for the communication with the provisioning API.
provisioningApi:
# -- Connection parameters
connection:
# -- The base URL the provisioning API is reachable at. (e.g. "https://
→provisioning-api")
baseUrl: ""
# -- Authentication parameters
auth:
# -- The username to authenticate with.
username: null
# -- The password to authenticate with.
password: null
existingSecret:
# -- The name of an existing Secret to use for retrieving the password
# to authenticate with the Provisioning API.
#
# "provisioningApi.auth.password" will be ignored if this value is set.
name: null
keyMapping:
# -- The key to retrieve the password from. Setting this value allows
→to use
# a key with a different name.
password: null
# -- Database resync on first startup only if database is empty.
resync:
# -- Enable the database resync on first startup only if database is empty.
enabled: true
auth:
# -- The admin username to authenticate with the Provisioning API
# for re-creating the ox-connector subscriber.
username: "admin"
# -- The admin password to authenticate with the Provisioning API.
password: null
existingSecret:
# -- The name of an existing Secret to use for retrieving the password
# to authenticate with the Provisioning API.
#
# "provisioningApi.resync.auth.password" will be ignored if this value
→

```

(continues on next page)

(continued from previous page)

```

→ is set.
    name: null
    keyMapping:
      # -- The key to retrieve the password from. Setting this value
→ allows to use
      # a key with a different name.
    password: null
...

```

2. Set the required values for the following settings.

For optional settings and their default values, see *Configuration for Nubus for Kubernetes* (page 26).

Section *openXchange* (page 28)

openXchange.domainName (page 28): The OX mail domain that the connector uses to generate email addresses.

openXchange.auth.password (page 28): *OX_MASTER_PASSWORD* (page 18)

openXchange.oxSmtpServer (page 29): *OX_SMTP_SERVER* (page 17)

openXchange.oxImapServer (page 29): *OX_IMAP_SERVER* (page 17)

openXchange.oxSoapServer (page 29): *OX_SOAP_SERVER* (page 17)

openXchange.oxDbConnectionString (page 28): The SQLAlchemy connection string for the database.

The connection string uses the following pattern:

```

postgresql+psycpg2://<DATABASE_USERNAME>:<PASSWORD>@<HOSTNAME>/<DATABASE_
→NAME>

```

Replace *<DATABASE_USERNAME>*, *<PASSWORD>*, *<HOSTNAME>*, and *<DATABASE_NAME>* with the values for your database connection.

Section *provisioningApi* (page 31)

provisioningApi.auth.username (page 31): The value of the name attribute in [Listing 3.7](#).

provisioningApi.auth.password (page 31): The value of the password attribute in [Listing 3.7](#).

provisioningApi.connection.baseUrl (page 31): The base URL for the *Provisioning API* in the *Provisioning Service*.

The URL points to the Kubernetes service for the *Provisioning API*.

Use the following URL format:

```
http://release-name-provisioning-api
```

Replace *release-name* with the Helm Chart release name of your Nubus for Kubernetes installation.

provisioningApi.resync.auth.password (page 31): The password for the administrative user of the *Provisioning API*. Use this setting if you leave *provisioningApi.resync.enabled* (page 31) set to true.

Important

Nubus for Kubernetes doesn't expose the *Provisioning API* outside the cluster for security reasons.

Tip

Using existing Kubernetes secrets

Instead of specifying passwords directly in the values file, you can use existing Kubernetes secrets. Set `openXchange.auth.existingSecret.*`, `provisioningApi.auth.existingSecret.*`, and `provisioningApi.resync.auth.existingSecret.*` to the names of the secrets that contain the credentials.

When you use existing secrets, the *OX Consumer* ignores the inline password values.

See also

README file for the OX Consumer¹⁸

for information about the available Helm Chart values and their default settings.

Access to Provisioning API endpoint¹⁹

in *Univention Nubus - Customization and Modification Manual* [5] for information about how to access the *Provisioning API* inside the Kubernetes cluster.

Engine Configuration - SQLAlchemy 2.0 Documentation²⁰

for information about the configuration of database connections.

3.2.5 Install the OX Consumer

To install the *OX Consumer* with the configuration in *Prepare the OX Consumer configuration* (page 11), follow these steps:

1. Select an *OX Consumer* version from the [OX Connector repository tags](#)²¹.
2. Replace the following placeholders in [Listing 3.11](#):
 - `<NAMESPACE>`: Kubernetes namespace for the *OX Consumer*. You may choose the same namespace as your Nubus for Kubernetes deployment.
 - `<OX_CONSUMER_RELEASE_NAME>`: Helm Chart release name for the *OX Consumer*.

Caution

The release name for the *OX Consumer* must be different from your Nubus for Kubernetes release name. You risk deleting your Nubus for Kubernetes installation, if you select the same release name.

- `<OX_CONSUMER_VERSION>`: OX Consumer Helm Chart version to install.

Danger

Use a different Helm release name for the *OX Consumer*. Don't reuse the Nubus release name. If both releases use the same name in the same namespace, Helm can delete the existing Nubus for Kubernetes installation.

3. Install the *OX Consumer* with the command in [Listing 3.11](#).

¹⁸ <https://github.com/univention/ox-connector/blob/ucs5.2/helm/ox-connector/README.md>

¹⁹ <https://docs.software-univention.de/nubus-customization/1.x/en/api/provisioning.html#customization-api-provisioning-endpoint-access>

²⁰ <https://docs.sqlalchemy.org/en/20/core/engines.html#postgresql>

²¹ <https://github.com/univention/ox-connector/tags>

Listing 3.11: Install the *OX Consumer* with Helm

```
$ export NAMESPACE_FOR_CONSUMER="<NAMESPACE>"
$ export RELEASE_NAME="<OX_CONSUMER_RELEASE_NAME>"
$ export VERSION="<OX_CONSUMER_VERSION>"

$ helm upgrade \
  "$RELEASE_NAME" \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  --install \
  oci://artifacts.software-univention.de/nubus/charts/ox-connector \
  --values ox-consumer-values.yaml \
  --version "$VERSION"
```

4. Verify that the Helm release installs successfully and that the *OX Consumer* workloads are ready.

CONFIGURATION REFERENCE

Use this section to configure the **OX Connector** after installation. The configuration topics describe app settings, UCS-specific variables, additional CA certificates, and structured logging for the connector.

4.1 Configuration for UCS

Use this reference to configure the **OX Connector** app on Nubus for UCS. The app settings define how the connector reaches *OX App Suite* and how it provisions users, groups, shared accounts, and related features. The UCR variables describe UCS-specific values that affect the connector configuration.

4.1.1 App settings

OX_SOAP_SERVER

Defines the *OX App Suite* server. Provide the protocol and the fully qualified domain name (FQDN), for example `https://ox-app-suite.example.com`.

[*OX_SOAP_SERVER*](#) (page 17) tells the **OX Connector** app in the container where to look for the *OX App Suite* system. The container must resolve the FQDN.

For secure HTTPS connections, the container needs to validate the certificate. If the *OX App Suite* instance uses a self-signed certificate or a certificate that the **OX Connector** container can't validate, the container needs the root certificate for validation. For information about how to add self-signed certificates, see [*Import additional CA certificates on UCS*](#) (page 25).

Required	Type	Initial value
Yes	String	<code>https://\$hostname.\$domainname</code>

OX_IMAP_SERVER

Defines the default IMAP server for new users, if not explicitly set on the user object.

Required	Type	Initial value
Yes	String	<code>imap://\$hostname.\$domainname:143</code>

OX_SMTP_SERVER

Defines the SMTP server for new users, if not explicitly set on the user object.

Required	Type	Initial value
Yes	String	<code>smtp://\$hostname.\$domainname:587</code>

DEFAULT_CONTEXT

Defines the default context for users. The **OX Connector** doesn't create `DEFAULT_CONTEXT` automatically. Before the **OX Connector** provisions the first user, ensure that the default context exists.

To create a context, see [Usage](#) (page 35).

Required	Type	Initial value
Yes	Integer	10

OX_LANGUAGE

Defines the default language for new users.

Required	Type	Initial value
Yes	String	de_DE

LOCAL_TIMEZONE

Defines the default time zone for new users.

Required	Type	Initial value
Yes	String	Europe/Berlin

OX_MASTER_ADMIN

Defines the username for the *OX App Suite* administrator account, also called the *OX Admin user*. This user can create, modify, and delete contexts. The user must already exist. The administrator defines the username for the *OX Admin user* during the installation of *OX App Suite*.

Required	Type	Initial value
Yes	String	oxadminmaster

OX_MASTER_PASSWORD

Defines the password for the *OX Admin user*.

Required	Type	Initial value
No	Password	N/A

OX_IMAP_LOGIN

Defines the value that *OX App Suite* uses to access the user's inbox. If this value is empty, *OX App Suite* sets it to the user's email address.

Required	Type	Initial value
No	String	N/A

If you use single sign-on (SSO), append an asterisk and the mail server master user to this variable. For Dovecot, set `OX_IMAP_LOGIN` to `'{*dovecotadmin}'`.

The **OX Connector** interprets the curly braces as a template. Empty braces use `primaryMailAddress`. To use another user attribute, enter the attribute name in the braces, for example `{username}`.

OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE

Defines the value that *OX App Suite* uses to access the functional account inbox. If this value is empty, the **OX Connector** sets it to a concatenation of the functional account LDAP entry UUID and the LDAP UID of the user.

This template can include the functional account entry UUID (`fa_entry_uuid`), the functional account email address (`fa_email_address`), and any Univention Directory Manager (UDM) property of the OX user, including the user's `entry_uuid` and `dn`. Enclose every UDM property used in this template in `{{ }}`, for example `{{fa_entry_uuid}}{{username}}`. You can optionally separate multiple values with other text.

Required	Type	Initial value
No	String	N/A

If you use the *OX App Suite* app from Univention App Center, you can leave this app setting empty. An empty value is equivalent to `{{fa_entry_uuid}}{{username}}`.

If an existing **OX Connector** installation used only the functional account entry UUID, set this app setting to `{{fa_entry_uuid}}`.

Listing 4.1 shows valid template values.

Listing 4.1: Valid functional account login template values

```
"{{fa_entry_uuid}}:{{entry_uuid}}"
"{{username}}+{{fa_entry_uuid}}+{{dn}}"
"{{fa_email_address}}*dovecotadmin"
```

`{{fa_entry_uuid}}:{{entry_uuid}}`

Concatenates the functional account entry UUID and the user UUID, separated by two colons.

`{{username}}+{{fa_entry_uuid}}+{{dn}}`

Concatenates the username, the functional account entry UUID, and the user DN, separated by plus signs.

`{{fa_email_address}}*dovecotadmin`

Concatenates the functional account email address and the string `*dovecotadmin`.

If you use single sign-on, append an asterisk and the mail server master user to this variable. For Dovecot, set `OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE` to `'{{fa_email_address}}*dovecotadmin'`. Listing 4.2 shows the resulting login value for the functional account.

Listing 4.2: Resulting functional account login value with SSO

```
myfunctional_account@maildomain.de*dovecotadmin
```

OX_USER_IDENTIFIER

Defines the UDM user property that *OX App Suite* uses as the unique user identifier. If this app setting isn't set, the **OX Connector** uses the `username` property by default.

For Nubus for Kubernetes, see [*openXchange.mappings.userIdentifier*](#) (page 28).

Required	Type	Initial value
No	String	N/A

Caution

Use only a mandatory UDM user property that contains one non-empty value. If you specify a UDM user property that contains an empty value or a list of values, the **OX Connector** enters an error state. To resolve the error, set a valid property.

OX_GROUP_IDENTIFIER

Defines the UDM group property that *OX App Suite* uses as the unique group identifier. If this app setting isn't set, the **OX Connector** uses the `name` property by default.

For Nubus for Kubernetes, see [openXchange.mappings.groupIdentifier](#) (page 28).

Required	Type	Initial value
No	String	N/A

Caution

Use only a mandatory UDM group property that contains one non-empty value. If you specify a UDM group property that contains an empty value or a list of values, the **OX Connector** enters an error state. To resolve the error, set a valid property.

OX_SHARED_ACCOUNT_IDENTIFIER

Defines the UDM shared account property that *OX App Suite* uses as the unique shared account identifier. If this app setting isn't set, the **OX Connector** uses the `name` property by default.

For Nubus for Kubernetes, see [openXchange.mappings.sharedAccountIdentifier](#) (page 28).

Required	Type	Initial value
No	String	N/A

Caution

Use only a mandatory UDM shared account property that contains one non-empty value. If you specify a UDM shared account property that contains an empty value or a list of values, the **OX Connector** enters an error state. To resolve the error, set a valid property.

OX_CONNECTOR_LOG_LEVEL

Defines the log level for the **OX Connector** app. If this app setting isn't set, the **OX Connector** uses `INFO` by default.

Required	Type	Initial value
No	String	INFO

OX_ENABLE_DEPUTY_PERMISSIONS

Enables the provisioning of OX deputy permissions. Administrators can then set, modify, or delete deputy permissions for users in the *Management UI*.

For example, administrators can grant `user01` the roles *Viewer*, *Editor*, and *Author* for the calendar and mail modules for `user02`. Also, `user01` can send email on behalf of `user02`.

The default value is `False`. To enable the feature, set the app setting [OX_ENABLE_DEPUTY_PERMISSIONS](#) (page 20) to `True`.

To show the feature in the *Management UI*, you must enable the UMC representation for the extended attribute after the app installation or after the configuration. Run the command in [Listing 4.3](#) on the [Primary Directory Node](#)²² or a [Backup Directory Node](#)²³.

Listing 4.3: Activate the UMC representation of the enabled deputy permission feature.

```
$ univention-directory-manager \
  settings/extended_attribute modify \
  --dn "cn=oxDeputyPermissionGivenTo,cn=open-xchange,cn=custom attributes,
  →cn=univention,$(ucr get ldap/base)" \
  --set disableUDMWeb="0"
```

Required	Type	Initial value
No	Boolean	False

Important

The *Deputy Permissions* feature requires *OX App Suite* version 8 or later.

Users can modify their deputy permissions in *OX App Suite*. Provisioning through the **OX Connector** app in Nubus for UCS overwrites these settings.

See also

Deputy permissions: Technical Documentation²⁴
for more information about OX deputy permissions.

OX_CONNECTOR_STOP_ON_ERROR

Changes how the **OX Connector** app handles synchronization errors. Set one of the following values:

True

Stop on any error. The app retries the failed action until it succeeds or an administrator resolves the error manually.

False

Continue with other queued actions. The app moves the failed action to the morgue. The failed action no longer interferes with the connector run, and an administrator can examine it later.

For information about managing provisioning tasks in Nubus for UCS, see [Manage provisioning tasks](#) (page 55).

Required	Type	Initial value
No	Boolean	True

4.1.2 UCR variables

ox/context/id

The app setting `DEFAULT_CONTEXT` (page 17) sets the value of the UCR variable `ox/context/id` (page 21).

²² <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-Primary-Directory-Node>

²³ <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-Backup-Directory-Node>

²⁴ https://documentation.open-xchange.com/8/middleware/permissions_and_capabilities/deputy_permission.html

When you install the **OX Connector** app, it creates the extended attribute `oxContext` and uses the value from `ox/context/id` (page 21) as the initial value for the extended attribute `oxContext`.

When an administrator creates a user account that the **OX Connector** app synchronizes, UDM sets the OX context for the user account to the value of the extended attribute `oxContext`.

Caution

The UCR variable `ox/context/id` (page 21) **isn't** for manual use.

Changing the variable **doesn't** change the OX context on existing user accounts.

Changing the value of the app setting `DEFAULT_CONTEXT` (page 17) changes **neither** `ox/context/id` (page 21) **nor** the extended attribute `oxContext`.

4.1.3 User attribute mapping

Since version 2.2.9, you can change the mapping between *Open-Xchange* and *UDM* properties. Use the `change_attribute_mapping.py` script from the app. The script creates a JSON file with the Open-Xchange property mapping and provisioning data.

Don't modify the file manually. Use only the script. The script stores the JSON file at `/var/lib/univention-appcenter/apps/ox-connector/data/AttributeMapping.json`.

If the file doesn't exist, the **OX Connector** app uses the default mapping. The default mapping comes from the following file inside the container of the app:

```
/usr/lib/python3.9/site-packages/univention/ox/provisioning/default_user_mapping.py.
```

The script supports these actions:

modify

Performs operations that change the current mapping.

restore_default

Restores the default mapping.

dump

Writes the current JSON mapping to the console.

The `modify` action supports these options:

modify --set

Changes the UDM property used to provision an Open-Xchange property. [Listing 4.4](#) shows how to map the Open-Xchange property `userfield01` to the UDM property `description`.

Listing 4.4: Set the mapping of an Open-Xchange property to a UDM property

```
$ python3 /var/lib/univention-appcenter/apps/ox-connector/data/resources/
↪change_attribute_mapping.py \
  modify \
    --set userfield01 description
```

You can use the `modify --set` (page 22) argument multiple times in the same invocation. [Listing 4.5](#) shows how to map multiple Open-Xchange properties to multiple UDM properties.

Listing 4.5: Set multiple Open-Xchange properties to multiple UDM properties

```
$ python3 /var/lib/univention-appcenter/apps/ox-connector/data/resources/
↪change_attribute_mapping.py \
```

(continues on next page)

(continued from previous page)

```

modify \
--set userfield01 description \
--set given_name custom_attribute

```

modify --unset

Removes the Open-Xchange property from the mapping if the property isn't marked as required. You can use this option to remove properties from synchronization.

Listing 4.6: Unset the OX property userfield01.

```

$ python3 /var/lib/univention-appcenter/apps/ox-connector/data/resources/
→change_attribute_mapping.py \
  modify \
  --unset userfield01

```

modify --set_alternatives

Sets alternative UDM properties for synchronization when the primary UDM property is None. [Listing 4.7](#) shows how to set the example attributes `CustomAttributeUserMail` and `CustomAttributeUserMail2` as alternatives to the Open-Xchange property `email1`.

Listing 4.7: Set example attributes as alternatives to an Open-Xchange property

```

$ python3 /var/lib/univention-appcenter/apps/ox-connector/data/resources/
→change_attribute_mapping.py \
  modify \
  --set_alternatives email1 CustomAttributeUserMail CustomAttributeUserMail2

```

modify --unset_alternatives

Removes the current alternatives for an OX property.

[Listing 4.8](#) shows how to remove the alternative attributes for the OX property `email1`.

Listing 4.8: Unset the alternative attributes for the OX property email1

```

$ python3 /var/lib/univention-appcenter/apps/ox-connector/data/resources/
→change_attribute_mapping.py \
  modify \
  --unset_alternatives email1

```

You can migrate an existing attribute mapping from the *OX App Suite* app in Univention App Center. To migrate the mapping, do the following:

1. On the Nubus for UCS system that runs *OX App Suite*, run the command in [Listing 4.9](#).

Listing 4.9: Create the migration command for an existing attribute mapping

```

$ python3 <<EOF
from univention.config_registry import ConfigRegistry
ucr = ConfigRegistry()
ucr.load()

changed_mapping_single = {
    'displayname': 'display_name',
    'givenname': 'given_name',
    'surname': 'sur_name',
    'categories': 'employee_type',

```

(continues on next page)

(continued from previous page)

```

    'quota': 'max_quota',
    }

changed_mapping_multi = {
    'telephone_business': ['telephone_business1', 'telephone_business2'],
    'telephone_home': ['telephone_home1', 'telephone_home2'],
}

ucr_ldap2ox = ucr.get('ox/listener/user/ldap/attributes/mapping/ldap2ox', '').
↳strip()
ucr_ldap2oxmulti = ucr.get('ox/listener/user/ldap/attributes/mapping/
↳ldap2oxmulti', '').strip()
command = []
if ucr_ldap2ox:
    for entry in ucr_ldap2ox.split():
        value, key = entry.split(':', 1)
        if value is None:
            command.append(f"--unset {changed_mapping_single.get(key, key)}")
        else:
            command.append(f"--set {changed_mapping_single.get(key, key)} {value}")

if ucr_ldap2oxmulti:
    ldap2oxmulti = {}
    for entry in ucr_ldap2oxmulti.split():
        value, key = entry.split(':', 1)
        if value is None:
            for v in changed_mapping_multi.get(key, [key]):
                command.append(f"--unset {v}")
        else:
            for v in changed_mapping_multi.get(key, [key]):
                command.append(f"--set {v} {value}")

if command:
    print("Run the following command on the ox-connector server to update_
↳attribute mapping:")
    print("python3 /var/lib/univention-appcenter/apps/ox-connector/data/
↳resources/change_attribute_mapping.py modify " + " ".join(command))
else:
    print("Nothing to do.")
EOF

```

2. Copy the generated command from the output.
3. On the Nubus for UCS system where the **OX Connector** app runs, run the generated command.
4. Verify the migration result. Run the command in [Listing 4.10](#) and check that the expected Open-Xchange properties map to the expected UDM properties.

Listing 4.10: Verify the migrated attribute mapping

```
$ python3 \
  /var/lib/univention-appcenter/apps/ox-connector/data/resources/change_
↪attribute_mapping.py \
  dump
```

4.2 Import additional CA certificates on UCS

The **OX Connector** app on Nubus for UCS runs in a container with its own CA store. By default, the app imports the Nubus for UCS root CA into this store. This import enables a secure connection to the Nubus LDAP directory. You might need additional CA certificates when provisioning to a remote *OX App Suite* installation.

Before you start, make sure that you meet the following prerequisites:

- You have installed the **OX Connector** app.
- You have the CA certificate files in PEM format.

Use these steps on the system that runs the **OX Connector** app:

1. Create the `/var/lib/univention-appcenter/apps/ox-connector/data/conf/ca-certificates/` directory.
2. Copy the CA certificate files in PEM format to this directory. Use only files with the `.pem` filename extension. [Listing 4.11](#) shows an example.

Listing 4.11: Examples of additional CA certificates

```
$ file /var/lib/univention-appcenter/apps/ox-connector/data/conf/ca-
↪certificates/*.pem
.../ox-connector/data/conf/ca-certificates/cert1.pem: PEM certificate
.../ox-connector/data/conf/ca-certificates/cert2.pem: PEM certificate
```

3. Run the command shown in [Listing 4.12](#) to reconfigure the **OX Connector** app. The command finishes without errors and adds the certificates to the app's certificate store. During app updates, the app also imports the certificates from the directory.

Listing 4.12: Manually reconfigure the OX Connector

```
$ univention-app configure ox-connector
```

4.3 Structured logging on UCS

Added in version 4.0.0.

The **OX Connector** app emits log messages in a structured format that follows the Nubus platform logging standard. Each log line contains a fixed set of fields followed by key-value pairs with context-specific data, as shown in [Listing 4.13](#).

Listing 4.13: Structured log line format

```
<timestamp> <severity:8> [<request ID>] <message>\t| <key=value ...> <source>
```

The fields have the following meanings:

timestamp

The timestamp is ISO 8601-formatted and includes sub-second precision and the time zone, for example `2024-06-01T12:00:00.123456+00:00`.

severity

The log level field has a width of 8 characters, for example INFO or WARNING. [OX_CONNECTOR_LOG_LEVEL](#) (page 20) controls this value.

request ID

The request ID is the sequence number of the provisioning message. It groups all log lines that belong to a single provisioning job. A dash (-) indicates that no job is active.

message

The message is the human-readable description of the event.

key=value pairs

The structured application data uses the `logfmt` format, for example `dn="uid=user01,dc=example,dc=com" object_type=users ms=42`.

source

The source is the Python module name that produced the log entry.

See also
Logging²⁵

in *Nubus Manual 1.x* [6] for information about the Nubus structured logging format, available log levels, and integration with log management systems.

4.4 Configuration for Nubus for Kubernetes

Use this reference to configure the **OX Connector** on Nubus for Kubernetes. This page uses the published **OX Connector** Helm Chart to provide the Helm Chart values reference.

4.4.1 Configure a database

Added in version v0.41.0: Add in *OX Connector* for Kubernetes version v.0.41.0.

To support the shared accounts feature, the *OX Connector* requires a PostgreSQL database to store account references. PostgreSQL is the only tested and supported database for the *OX Connector*. The database isn't part of the OX App Suite packaged and the OX Connector integration. You need to provide it separately.

Before you install or upgrade to a version $\geq 0.41.0$, ensure the following requirements for the database for the *OX Connector* in PostgreSQL:

- You have created an empty database for the *OX Connector*.
- You have created a dedicated database user for the *OX Connector* with the `ALL PRIVILEGES` privilege.

For information about how to configure the database connection, see [openXchange.oxDbConnectionString](#) (page 28) in *Prepare the OX Consumer configuration* (page 11).

4.4.2 Helm Chart references values

Name

`ox-connector`

Version

`0.42.1`

Description

A Helm chart for the `ox-connector`

You find the configuration options for `ox-connector` in the following sections.

²⁵ <https://docs.software-univention.de/nubus-manual/1.x/en/logging.html#nubus-logging>

4.4.3 affinity

affinity

#Global values

Default value: {}

4.4.4 environment

environment

Default value: {}

4.4.5 extraVolumeMounts

extraVolumeMounts

Optionally specify an extra list of additional volumeMounts.

Default value: []

4.4.6 extraVolumes

extraVolumes

Optionally specify an extra list of additional volumes.

Default value: []

4.4.7 fullnameOverride

fullnameOverride

Default value: ""

4.4.8 global

global.imagePullPolicy

Define an ImagePullPolicy. # Ref.: <https://kubernetes.io/docs/concepts/containers/images/#image-pull-policy> # "IfNotPresent" => The image is pulled only if it is not already present locally. "Always" => Every time the kubelet launches a container, the kubelet queries the container image registry to resolve the name to an image digest. If the kubelet has a container image with that exact digest cached locally, the kubelet uses its cached image; otherwise, the kubelet pulls the image with the resolved digest, and uses that image to launch the container. "Never" => The kubelet does not try fetching the image. If the image is somehow already present locally, the kubelet attempts to start the container; otherwise, startup fails.

Default value: null

global.imagePullSecrets

Credentials to fetch images from private registry. Ref: <https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/> # imagePullSecrets: - "docker-registry"

Default value: []

global.imageRegistry

Container registry address.

Default value:

"artifacts.software-univention.de"

4.4.9 `nameOverride`

`nameOverride`

Default value: ""

4.4.10 `nodeSelector`

`nodeSelector`

Default value: {}

4.4.11 `openXchange`

`openXchange.auth.existingSecret.keyMapping.password`

The key to retrieve the password from. Setting this value allows to use a key with a different name.

Default value: `null`

`openXchange.auth.existingSecret.name`

The name of an existing Secret to use for retrieving the password for the ox admin password. # “oxConnector.auth.password” will be ignored if this value is set.

Default value: `null`

`openXchange.auth.password`

OX Admin password

Default value: `null`

`openXchange.auth.username`

OX Admin username (the OX Admin can create, modify, delete contexts; has to exist)

Default value: `"oxadminmaster"`

`openXchange.domainName`

OX-Mail-Domain to generate OX-email-addresses

Default value: `null`

`openXchange.logLevel`

OX Connector log level Chose from “DEBUG”, “INFO”, “WARNING” and “ERROR”.

Default value: `"INFO"`

`openXchange.mappings.groupIdentifier`

UDM group property that is used as the unique group identifier for OX

Default value: `"name"`

`openXchange.mappings.sharedAccountIdentifier`

UDM shared account property that is used as the unique account identifier for OX

Default value: `"name"`

`openXchange.mappings.userIdentifier`

UDM user property that is used as the unique user identifier for OX

Default value: `"username"`

`openXchange.oxDbConnectionString`

SQLAlchemy DB connection URL for the OX connector.

Default value: `null`

openXchange.oxDefaultContext

Default context for users (has to exist)

Default value: "10"**openXchange.oxDeputyPermissions**

Ox Connector deputy permissions flag.

Default value: false**openXchange.oxImapServer**

Default IMAP server for new users (if not set explicitly there)

Default value: null**openXchange.oxLanguage**

Default language for new users

Default value: "de_DE"**openXchange.oxLocalTimezone**

Default timezone for new users

Default value: "Europe/Berlin"**openXchange.oxSharedAccount**

Ox Connector shared account flag.

Default value: true**openXchange.oxSmtptServer**

Default SMTP server for new users (if not set explicitly there)

Default value: null**openXchange.oxSoapServer**

The server where Open-Xchange is installed

Default value: null

4.4.12 oxConnector

oxConnector.extraEnvVars

Array with extra environment variables to add to containers. # extraEnvVars: - name: FOO value: "bar"

Default value: []**oxConnector.image.pullPolicy****Default value:** null**oxConnector.image.registry****Default value:** null**oxConnector.image.repository****Default value:**`"nubus/images/ox-connector-standalone"`**oxConnector.image.tag****Default value:**`"0.42.1@sha256:16a17bf4d2ca2074d16ec708798012c9543091028cd3bc5a10e1c8be4fb14a64"`
→

4.4.13 persistence

persistence.size

Specify PVCs size

Default value: "1Gi"

persistence.storageClass

Specify storageClassName - Leave empty to use the default storage class

Default value: ""

4.4.14 podAnnotations

podAnnotations

Default value: {}

4.4.15 podSecurityContext

podSecurityContext.fsGroup

Default value: 1000

podSecurityContext.runAsGroup

Default value: 1000

podSecurityContext.runAsNonRoot

Default value: true

podSecurityContext.runAsUser

Default value: 1000

podSecurityContext.seccompProfile.type

Default value: "RuntimeDefault"

4.4.16 probes

probes.liveness.exec.command

Default value: ["/bin/sh", "-c", "exit 0\n"]

probes.liveness.failureThreshold

Default value: 3

probes.liveness.initialDelaySeconds

Default value: 120

probes.liveness.periodSeconds

Default value: 30

probes.liveness.successThreshold

Default value: 1

probes.liveness.timeoutSeconds

Default value: 3

probes.readiness.exec.command

Default value: ["/bin/sh", "-c", "exit 0\n"]

probes.readiness.failureThreshold

Default value: 30

probes.readiness.initialDelaySeconds

Default value: 30

`probes.readiness.periodSeconds`

Default value: 15

`probes.readiness.successThreshold`

Default value: 1

`probes.readiness.timeoutSeconds`

Default value: 3

4.4.17 provisioningApi

`provisioningApi.auth.existingSecret.keyMapping.password`

The key to retrieve the password from. Setting this value allows to use a key with a different name.

Default value: null

`provisioningApi.auth.existingSecret.name`

The name of an existing Secret to use for retrieving the password to authenticate with the Provisioning API. #
“provisioningApi.auth.password” will be ignored if this value is set.

Default value: null

`provisioningApi.auth.password`

The password to authenticate with.

Default value: null

`provisioningApi.auth.username`

The username to authenticate with.

Default value: "ox-consumer"

`provisioningApi.config.maxAcknowledgementRetries`

The maximum number of retries for acknowledging a message

Default value: 3

`provisioningApi.connection.baseUrl`

The base URL the provisioning API is reachable at. (e.g. “https://provisioning-api”)

Default value: ""

`provisioningApi.resync.auth.existingSecret.keyMapping.password`

The key to retrieve the password from. Setting this value allows to use a key with a different name.

Default value: null

`provisioningApi.resync.auth.existingSecret.name`

The name of an existing Secret to use for retrieving the password to authenticate with the Provisioning API. #
“provisioningApi.resync.auth.password” will be ignored if this value is set.

Default value: null

`provisioningApi.resync.auth.password`

The admin password to authenticate with the Provisioning API.

Default value: null

`provisioningApi.resync.auth.username`

The admin username to authenticate with the Provisioning API for recreating the ox-connector subscriber.

Default value: "admin"

`provisioningApi.resync.enabled`

Enable the database resync on first startup only if database is empty.

Default value: true

4.4.18 replicaCount

`replicaCount`

Default value: 1

4.4.19 resources

`resources.limits.cpu`

Default value: "4"

`resources.limits.memory`

Default value: "4Gi"

`resources.requests.cpu`

Default value: "250m"

`resources.requests.memory`

Default value: "512Mi"

4.4.20 resourcesWaitForDependency

`resourcesWaitForDependency.limits.cpu`

Default value: "200m"

`resourcesWaitForDependency.limits.memory`

Default value: "128Mi"

`resourcesWaitForDependency.requests.cpu`

Default value: "100m"

`resourcesWaitForDependency.requests.memory`

Default value: "64Mi"

4.4.21 securityContext

`securityContext.allowPrivilegeEscalation`

Default value: false

`securityContext.capabilities.drop`

Default value: ["ALL"]

`securityContext.privileged`

Default value: false

`securityContext.readOnlyRootFilesystem`

Default value: true

`securityContext.runAsGroup`

Default value: 1000

`securityContext.runAsNonRoot`

Default value: true

`securityContext.runAsUser`

Default value: 1000

`securityContext.seccompProfile.type`

Default value: "RuntimeDefault"

4.4.22 serviceAccount

serviceAccount.annotations

Annotations to add to the service account

Default value: {}

serviceAccount.automountServiceAccountToken

#@param serviceAccount.automountServiceAccountToken Allows auto mount of ServiceAccountToken on the serviceAccount created #Can be set to false if pods using this serviceAccount do not need to use K8s API ##

Default value: false

serviceAccount.create

Specifies whether a service account should be created

Default value: true

serviceAccount.labels

Additional custom labels for the ServiceAccount.

Default value: {}

serviceAccount.name

The name of the service account to use. If not set and create is true, a name is generated using the fullname template

Default value: ""

4.4.23 tolerations

tolerations

Default value: []

4.4.24 waitForDependency

waitForDependency.image.pullPolicy

Default value: null

waitForDependency.image.registry

Default value: null

waitForDependency.image.repository

Default value:

```
"nubus/images/wait-for-dependency"
```

waitForDependency.image.tag

Default value:

```
"0.36.
→12@sha256:7150d72c8f342a05b945ce1b21464864aa91590d00f65ebe4b628571cce34efc"
```


USAGE

Use the following sections to manage OX App Suite provisioning objects through the *Management UI*. They cover contexts, users, groups, access profiles, functional accounts, resources, shared accounts, and migration from functional accounts.

5.1 Manage OX objects

Use this page to manage OX App Suite objects. The OX Connector synchronizes these objects from the LDAP directory. The following instructions explain how to create and configure contexts, user accounts, groups, access profiles, functional accounts, and resources in the *Management UI*. Read them to control which directory objects are available in OX App Suite and how the connector provisions them.

To complete the following tasks, sign in to the *Management UI* with a user account that has domain administration rights.

See also

Management UI²⁶

in *Nubus Manual 1.x* [6] for information about the *Management UI*.

Sign-in²⁷

in *Nubus Manual 1.x* [6] for information about sign-in.

Delegated administration for management modules²⁸

in *Nubus for UCS 5.2 - Operation Manual* [1] for information about delegative administration.

5.1.1 Contexts

OX App Suite uses *contexts* to collect users, groups, and resources for collaboration in a virtual space. Data from one context isn't visible to other contexts. For more information about contexts, see *App Suite Context management* [7].

To view, add, update, or delete a context, navigate to *Domain* ▶ *OX Contexts* in the *Management UI*.

5.1.2 Users

To add a user to OX App Suite, create a user account or update an existing one.

Add user account

To create a user account:

1. Navigate to *Users* ▶ *Users* in the *Management UI*.
2. Select *Add* and the *User template* `open-xchange groupware` account.

²⁶ <https://docs.software-univention.de/nubus-manual/1.x/en/ui.html#nubus-ui>

²⁷ <https://docs.software-univention.de/nubus-manual/1.x/en/auth.html#nubus-authentication-sign-in>

²⁸ <https://docs.software-univention.de/ucs-operation/5.2/en/management-interface/delegated-administration.html#management-interface-delegated-administration>

3. Select *Next*.
4. Enter the required information. To enter additional attributes, select *Advanced*.
5. Select *Create user*.

Update user account

To update a user account:

1. Navigate to *Users* ▶ *Users* in the *Management UI*.
2. Select the username of the user account that you want to update.
3. On the *Apps* tab, select the *Open-Xchange* checkbox. The *Open-Xchange* tab appears.
4. Enter the user's primary email address at *General* ▶ *Primary email address (mailbox)*.
5. Select *Save*.

See also

User management²⁹

in *Nubus Manual 1.x* [6] for information about user management in Nubus.

5.1.3 Groups

The **OX Connector** app adds a group to the same context as the group members. When the last group member leaves the group, the connector removes the group from OX App Suite.

To enable a group for OX App Suite, follow these procedures:

Add group

To create a group:

1. Navigate to *Users* ▶ *Groups* in the *Management UI*.
2. Select *Add*.
3. On the *General* tab, enter the required information and add users as group members.
4. On the *OX App Suite* tab, select the *Activate Group in OX* checkbox.
5. Select *Create group*.

Update group

To update a group:

1. Navigate to *Users* ▶ *Groups* in the *Management UI*.
2. Select the group that you want to edit.
3. The *Groups* module in the *Management UI* automatically selects the *Activate Group in OX* checkbox when you edit a group. The *Management UI* displays a notification.
4. To prevent OX provisioning, clear the *Activate Group in OX* checkbox on the *OX App Suite* tab.
5. Select *Save*.

²⁹ <https://docs.software-univention.de/nubus-manual/1.x/en/users/index.html#nubus-user-management>

Warning

If you clear the *Activate Group in OX* checkbox while updating a group that exists in OX App Suite, the connector removes the group from OX App Suite.

To update a group from the command line, run the command in [Listing 5.1](#). Replace `GROUP_DN` with the distinguished name (DN) of the group.

Listing 5.1: Update a group from the command line

```
$ udm groups/group modify --dn GROUP_DN --set isOxGroup=OK
```

Remove group

To remove a group from OX App Suite:

1. Navigate to *Users* ▶ *Groups* in the *Management UI*.
2. Select the group that you want to edit.
3. On the *OX App Suite* tab, clear the *Activate Group in OX* checkbox.
4. Select *Save*.

To remove a group from OX App Suite, run the command in [Listing 5.2](#). Replace `GROUP_DN` with the distinguished name (DN) of the group.

Listing 5.2: Remove a group from the command line

```
$ udm groups/group modify --dn GROUP_DN --set isOxGroup=Not
```

See also**Group management³⁰**

in *Nubus Manual 1.x* [6] for information about group management in Nubus.

5.1.4 Access profiles

The OX Connector provides *access profiles* for OX App Suite users. To view custom *access profiles*:

1. Navigate to *Domain* ▶ *LDAP directory* in the *Management UI*.
2. In the *LDAP directory* module, open `open-xchange/accessprofiles/`.

For limitations about plausibility verification, see [No validation of access profile rights](#) (page 54).

5.1.5 Functional accounts

Deprecated since version 3.2.0: Open-Xchange deprecated this feature in favor of [Shared accounts](#) (page 38).

Added in version 2.0.0.

Functional accounts let users in the same context share a functional mailbox and its read status.

Use the `oxmail/functional_account` management module to add, update, or delete functional-account objects. When administrators grant a user permission, emails sent to functional-account addresses appear in the OX Mail view for that user.

Added in version 2.2.12.

³⁰ <https://docs.software-univention.de/nubus-manual/1.x/en/groups.html#nubus-groups>

The default directory position for `oxmail/functional_account` objects is: `cn=functional_accounts, cn=open-xchange, $LDAP_BASE`. Replace `$LDAP_BASE` with the LDAP base DN.

You can add default containers for `oxmail/functional_account`. The *Management UI* then prompts you to select a position when you create an object.

To add default containers:

1. In the *LDAP directory* module, open the `univention` container.
2. Open the `default_containers` object.
3. Select `OX App Suite`.
4. Add containers to the `Default container for OX functional accounts` list.

Enter the distinguished names (DNs) of existing container objects. Each DN must include the LDAP base DN.

5.1.6 Resources

Use *OX Resources* to manage bookable rooms and equipment. For more information about resource management, see *App Suite Resource management* [8].

To view resources, navigate to *Domain* ▶ *OX Resources* in the *Management UI*.

5.2 Shared accounts

Added in version 3.2.0: Added *OX Connector App* version 3.2.0.

Added in version v0.41.0: Added in *OX Connector* for Kubernetes version v.0.41.0.

OX App Suite lets users and groups access shared accounts. Users with access to a shared account can read its email and calendar entries. As an administrator, you can configure fine-grained permissions for users and groups. The OX Connector app provides management modules for the *Management UI* in Nubus that let you manage shared accounts and user and group permissions.

Important

The *Shared accounts* feature requires *OX App Suite* version 8.49 or later. A runtime check deactivates the feature when *OX App Suite* doesn't support shared accounts.

See also

Shared accounts³¹

in *Middleware Documentation* [9] for information about shared accounts in OX App Suite.

5.2.1 Management module for shared accounts

Use the `oxmail/shared_account` management module to manage shared accounts and their permissions. In the *Management UI*, find the management module under *LDAP directory* at `open-xchange/shared_account`.

Every `oxmail/shared_account` object contains a list of users and groups with their respective permissions. Each user and group entry in the list links to an `oxmail/shared_account_permission` object.

See also

LDAP directory module^{Page 39, 32}

in *Nubus Manual 1.x* [6] for information about the *LDAP directory* management module.

³¹ https://documentation.open-xchange.com/8/middleware/permissions_and_capabilities/shared_accounts.html

5.2.2 Management module for permissions

OX App Suite uses permission objects to control user and group access to shared accounts. OX Connector provides the following predefined *permissions* for OX App Suite shared accounts:

- *Full Calendar Access*
- *Full Mail Access*
- *Full Mail and Calendar Access*
- *Read-Only Mail Access*

You can also create permissions to meet your requirements.

Use the `oxmail/shared_account_permission` management module to manage permissions for shared accounts. In the *Management UI*, find the management module under *LDAP directory* at `open-xchange/shared_account_permission`.

When you create an `oxmail/shared_account` object, you can grant permissions to users and groups in the *Management UI*.

5.2.3 Migration from functional accounts to shared accounts

Added in version 3.2.1.

OX App Suite has deprecated *Functional accounts* (page 37). OX Connector provides a script that lets you migrate functional accounts to shared accounts.

Before you run the script, **Dovecot** must use the email address as the unique identifier for the mail accounts.

Danger

If your **Dovecot** installation uses a unique identifier other than the email address, **don't run** the migration script. In that case, the script deletes your functional accounts and creates shared accounts without their content.

Before you use the migration script in production, run it with the `--dry-run` option and review the output. The `--dry-run` option doesn't make changes. It displays the migration steps that the script would perform.

To view the migration script parameters, run the script with the `--help` option. The help output describes migrating multiple functional accounts in one run. It also explains how to supply credentials through environment variables.

For troubleshooting, see *Troubleshoot migration from functional accounts to shared accounts* (page 73).

Choose the instructions for your OX Connector deployment.

Nubus for UCS

For Nubus for UCS, run the migration script on the system where you installed the OX Connector. Use the commands in [Listing 5.3](#) and [Listing 5.4](#). In the following command listings, replace the placeholders with values for these inputs:

UDM_USERNAME

The username for the Univention Directory Manager (UDM) user account. The user account must be a member of the [Authorization groups](#)³² for the *UDM HTTP REST API*.

UDM_PASSWORD

The password for the `UDM_USERNAME`.

REST_API_HOSTNAME

The fully qualified domain name (FQDN) of the UDM HTTP REST API in your domain.

³² <https://docs.software-univention.de/nubus-manual/1.x/en/management/domain/ldap.html#nubus-domain-ldap>

³³ <https://docs.software-univention.de/nubus-customization/1.x/en/api/udm-rest.html#customization-api-udm-rest-auth-group>

OPTIONAL_DESTINATION

The optional full distinguished name (DN) for the new shared account when migrating one functional account. For example, `cn=example,cn=shared_accounts,cn=open-xchange,LDAP_BASE`. If you omit this value, the script derives the destination DN.

DESTINATION_OX_CONTEXT

The OX Context where the migration script creates the shared account.

Listing 5.3: Prepare migration to shared accounts

```
$ export UDM_USERNAME="MANAGEMENT_USERNAME"
$ export UDM_PASSWORD="MANAGEMENT_PASSWORD"
$ export REST_API_HOSTNAME="MANAGEMENT_API_HOSTNAME"
$ export OPTIONAL_DESTINATION="OPTIONAL_DESTINATION_DN"
$ export DESTINATION_OX_CONTEXT=OX_CONTEXT_ID
```

Listing 5.4: Test the migration from functional accounts to shared accounts

```
$ univention-app shell \
  ox-connector \
  /usr/local/share/ox-connector/resources/migrate_fupo_to_shared_account.py \
  "cn=example_fupo,cn=functional_accounts,cn=open-xchange,$(ucr get ldap/base)" \
  "Full Mail Access" \
  "$OPTIONAL_DESTINATION" \
  "$UDM_USERNAME" \
  "$UDM_PASSWORD" \
  "https://$REST_API_HOSTNAME/univention/udm" \
  --ox-context $DESTINATION_OX_CONTEXT \
  --dry-run
```

After you verify the output, remove `--dry-run` and rerun the command to migrate the functional account. Confirm that the output contains `Resulting Shared Account:` and review the printed shared-account properties.

Nubus for Kubernetes

To run the migration script in Nubus for Kubernetes, follow these steps.

1. To set the environment variables for your Nubus for Kubernetes environment and OX Consumer deployment, run the commands in [Listing 5.5](#).

NAMESPACE_N4K

The Kubernetes namespace for your Nubus for Kubernetes deployment.

RELEASE_N4K

The release name for your Nubus for Kubernetes deployment. To list the release names in your namespace, run the command in [Listing 5.6](#).

NAMESPACE_CONNECTOR

The Kubernetes namespace of your OX Connector deployment. Typically, it is the same namespace as your Nubus for Kubernetes deployment.

Listing 5.5: Set environment variables for Nubus for Kubernetes and the OX Connector

```
$ export NAMESPACE_CONNECTOR="CONNECTOR_NAMESPACE"
$ export NAMESPACE_N4K="NUBUS_NAMESPACE"
$ export RELEASE_N4K="NUBUS_RELEASE"
```

Listing 5.6: Show the release names in the namespace of Nubus for Kubernetes

```
$ helm --namespace "$NAMESPACE_N4K" list -q
```

2. Retrieve the LDAP base DN from your Nubus for Kubernetes environment.

You need the LDAP base DN of your Nubus for Kubernetes deployment. You set the LDAP base DN in the `custom_values.yaml` file during the [deployment of Nubus for Kubernetes](#)³⁴. To retrieve the LDAP base DN, run the command in [Listing 5.7](#).

LDAP_BASE

The LDAP base DN of your directory service.

Listing 5.7: Retrieve the LDAP base DN from Nubus for Kubernetes

```
$ export LDAP_BASE="$(kubectl \
  --namespace "$NAMESPACE_N4K" \
  get configmap \
  "$RELEASE_N4K-ldap-server" \
  -o "jsonpath={.data.LDAP_BASE_DN}")"
```

3. Configure the remaining environment variables for the migration script.

UDM_USERNAME

The username for the UDM user. The user account must be a member of the [Authorization groups](#)³⁵ for the *UDM HTTP REST API*.

UDM_PASSWORD

The password for the UDM_USERNAME.

FUNCTIONAL_ACCOUNT

The LDAP distinguished name (DN) of the functional account that you want to migrate.

DESTINATION_OX_CONTEXT

The OX Context where the migration script creates the shared account.

Listing 5.8: Define the remaining parameters for the migration

```
$ export UDM_USERNAME="MANAGEMENT_USERNAME"
$ export UDM_PASSWORD="MANAGEMENT_PASSWORD"
$ export FUNCTIONAL_ACCOUNT="cn=example_fupo,cn=functional_accounts,cn=open-
  ↪xchange,$LDAP_BASE"
$ export DESTINATION_OX_CONTEXT=OX_CONTEXT_ID
```

4. Test the migration script. The example command in [Listing 5.9](#) uses the variables that you defined in the previous steps. Run the command with the `--dry-run` option.

If the output is as expected, remove `--dry-run` and rerun the command. Confirm that the output contains `Resulting Shared Account:` and review the printed shared-account properties before you use the script in production.

Listing 5.9: Test the migration script

```
$ kubectl \
  --namespace="$NAMESPACE_CONNECTOR" \
  get pods

$ export CONNECTOR_POD="OX_CONNECTOR_POD"
```

(continues on next page)

³⁴ <https://docs.software-univention.de/nubus-kubernetes-operation/1.x/en/deploy.html#nubus-deployment-all-deps>

³⁵ <https://docs.software-univention.de/nubus-customization/1.x/en/api/udm-rest.html#customization-api-udm-rest-auth-group>

(continued from previous page)

```
$ kubectl \
  --namespace="$NAMESPACE_CONNECTOR" \
  exec "$CONNECTOR_POD" -c main -- /bin/bash \
  -c 'UDM_USERNAME="$UDM_USERNAME" \
  UDM_URL="http://nubus-udm-rest-api:9979/univention/udm/" \
  UDM_PASSWORD="$UDM_PASSWORD" python3 \
  /usr/local/share/ox-connector/resources/migrate_fupo_to_shared_account.py \
  "$FUNCTIONAL_ACCOUNT" \
  "Full Mail Access" \
  --ox-context "$DESTINATION_OX_CONTEXT" \
  --dry-run'
```

ARCHITECTURE

Note

This chapter uses the *ArchiMate*® enterprise architecture modeling notation to visualize the OX Connector architecture. For more information, see *ArchiMate*³⁶ in *Univention Corporate Server Architecture* [10].

ArchiMate® is a registered trademark of The Open Group. The diagrams are independently created by Univention GmbH and aren't endorsed or certified by The Open Group.

6.1 Shared architecture

This page describes architecture components and connector behavior common to Nubus for UCS and Nubus for Kubernetes.

For deployment-specific details, see *For Nubus for UCS* (page 44) and *For Nubus for Kubernetes* (page 48).

6.1.1 Common architecture components

The OX Connector has the following common architecture components:

Identity Store and Directory Service

The *Identity Store and Directory Service* includes the OpenLDAP software that provides the LDAP directory in Nubus. The LDAP directory stores identity and infrastructure data in the domain. For more information, see *LDAP directory service*³⁷ in *Nubus for UCS 5.2 - Operation Manual* [1].

Directory Manager

The *Directory Manager* is an abstraction layer of directory objects in the *Identity Store and Directory Service*. For more information, see *Directory Manager*³⁸ in *Nubus for Kubernetes - Architecture Manual 1.x* [11].

Provisioning Service

The OX Connector subscribes to the *Provisioning Service* through the *Provisioning API* for the Univention Directory Manager (UDM) modules that it provisions. For information about change processing, consumer registration, and *prefill*, see *Provisioning Service*³⁹ in *Nubus for Kubernetes - Architecture Manual 1.x* [11].

Provisioning API

The OX Connector subscribes through the *Provisioning API* for the UDM modules that it provisions. For information about the API and consumer registration, see *Consumer Registration HTTP REST API*⁴⁰ in *Nubus for Kubernetes - Architecture Manual 1.x* [11].

OX Connector

OX Connector connects Nubus identity management to OX App Suite. The connector receives data about

³⁶ <https://docs.software-univention.de/architecture/5.2/en/appendix/architecture-notation.html#architecture-notation-archimate>

³⁷ <https://docs.software-univention.de/ucs-operation/5.2/en/domain-infrastructure/ldap-directory.html#domain-infrastructure-ldap-directory>

³⁸ [https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/directory-manager.html#](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/directory-manager.html#component-directory-manager)

[component-directory-manager](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/directory-manager.html#component-directory-manager)

³⁹ [https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#component-provisioning-service)

[component-provisioning-service](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#component-provisioning-service)

⁴⁰ [https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#component-provisioning-service-consumer-registration-http-rest-api)

[component-provisioning-service-consumer-registration-http-rest-api](https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/components/provisioning-service.html#component-provisioning-service-consumer-registration-http-rest-api)

changes in the LDAP directory. A *OX Connector Provisioning Consumer* handles the data, processes it, and sends it to the *SOAP API* in OX App Suite.

OX Connector Provisioning Consumer

The *OX Connector Provisioning Consumer* is the *OX Connector* component that receives provisioning messages, stores work in the connector database, and provisions *OX App Suite* objects.

OX App Suite

OX App Suite is the groupware and collaboration software from Open-Xchange.

SOAP API

OX App Suite provides a *SOAP API* for receiving data and running remote procedure calls. The connector uses the *SOAP API* to create, update, or delete *OX App Suite* objects.

6.1.2 Provisioned attributes

The OX Connector maps UDM attributes to OX App Suite attributes. The default user attribute mapping provisions the following attributes:

- User names, email addresses, and aliases
- Business and personal contact details
- Organization and address data
- Profile images and dates
- Mailbox quotas
- IMAP and SMTP servers
- User-defined fields

6.1.3 Database of object state

OX App Suite assigns an *internal ID* to each object that it creates or updates. After successfully processing an object, the connector stores its state, including the internal ID, in a local database. The connector doesn't store the internal ID in the LDAP directory.

The connector uses stored internal IDs for operations that require them. For example, when it updates a group through the *SOAP API*, the request must include the internal ID of each group member. This avoids a remote lookup in OX App Suite for each group member.

6.2 For Nubus for UCS

The **OX Connector** app architecture includes the following elements:

- The Nubus for UCS operating environment includes the App Center and Docker Engine. A container runs OX Connector.
- The container image contains the OX Connector software.
- The OpenLDAP directory in Nubus for UCS is the identity management source for OX App Suite.

Before you continue reading, ensure you know *Shared architecture* (page 43).

6.2.1 Overview

The **OX Connector** uses the software in its container image to provision identity data from Nubus for UCS to OX App Suite. The OX Connector connects to the OX App Suite SOAP API. It creates, updates, or deletes OX App Suite objects, including users, groups, contexts, and resources, in response to relevant LDAP directory changes. Fig. 6.4 shows the following components, their boundaries, and dependencies:

1. *Identity Store and Directory Service*
2. *Directory Manager*

3. *Provisioning Service*
4. *Provisioning API*
5. *OX Connector Provisioning Consumer*
6. *OX App Suite*

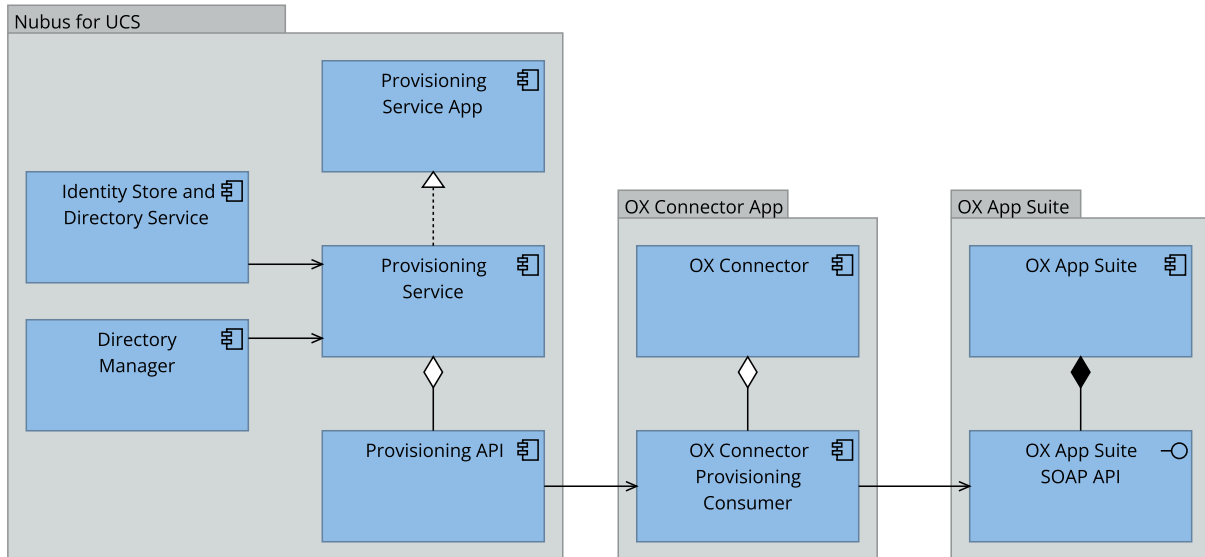


Fig. 6.1: OX Connector app architecture

Detailed OX Connector app processing, click to open.

Fig. 6.2 includes the *Provisioning Service* with all its parts and what happens there before the data moves to the *OX Connector*. Follow the *Flow* relations from the top right corner to the OX App Suite.

6.2.2 How the connector works

Univention Directory Manager (UDM) is an object layer on top of the LDAP directory in UCS. The OX Connector reacts to changes in the following UDM modules:

- Source modules:
 - users/user
 - groups/group
- Connector-specific modules:
 - oxmail/oxcontext
 - oxresources/oxresources
 - oxmail/accessprofile
 - oxmail/functional_account
 - oxmail/shared_account
 - oxmail/shared_account_permission

The connector processes changes to these modules and sends data to the SOAP API in OX App Suite when required.

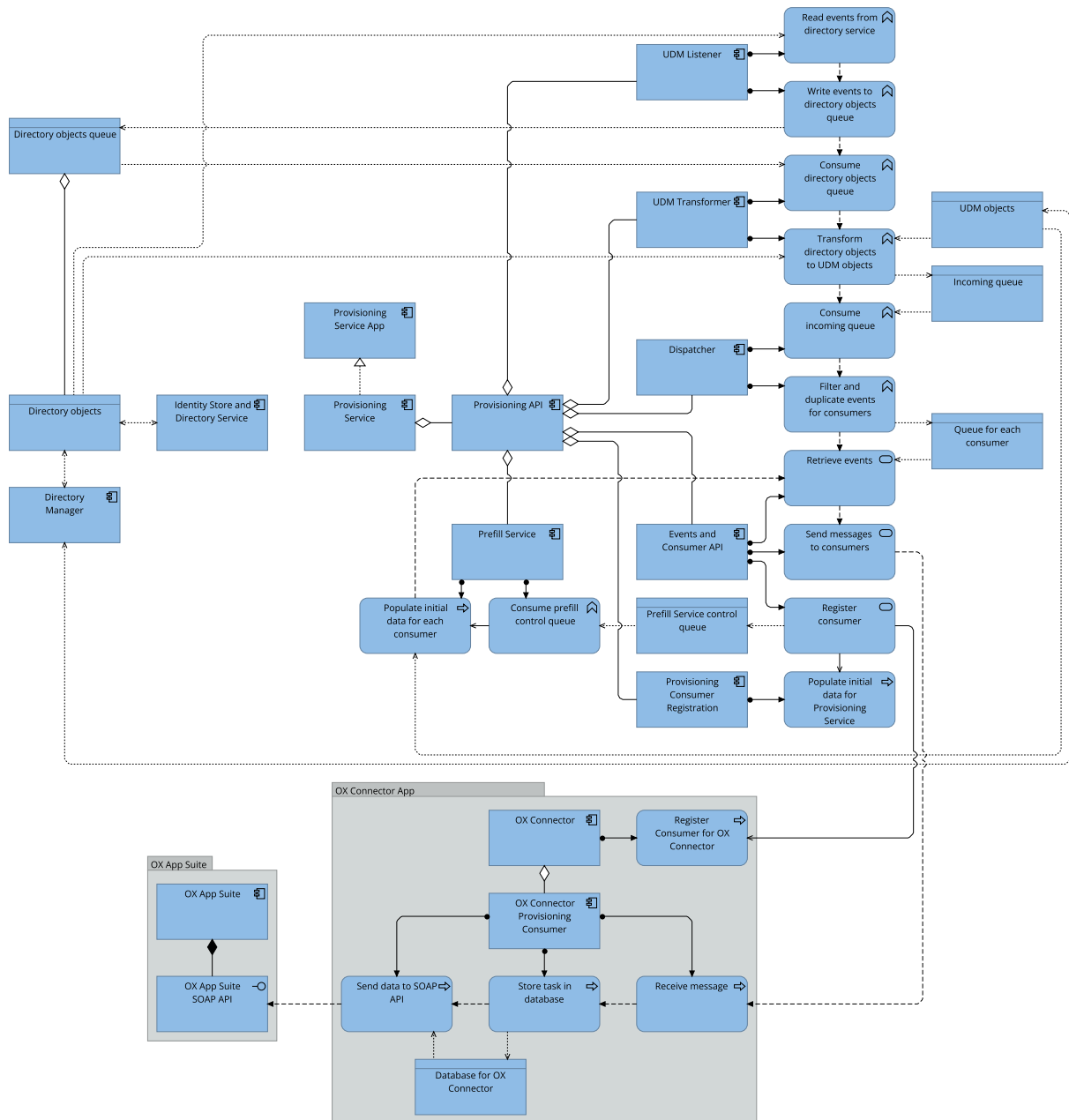


Fig. 6.2: OX Connector app architecture in more detail
Click the diagram to zoom in.

Access profiles

When the `oxmail/accessprofile` UDM module changes, the connector rewrites the local file `/var/lib/univention-appcenter/apps/ox-connector/data/ModuleAccessDefinitions.properties`. It doesn't send access-profile data directly to the SOAP API in OX App Suite. When the connector provisions a user, it applies the access-profile definitions from this file through the SOAP API.

Administrators can find *access profiles* in the *Management UI*, in the *LDAP directory* module, at *open-xchange* ▶ *accessprofile*.

Provisioning

For a visualization of the provisioning process, see Fig. 6.3. The following steps describe how the OX Connector provisions changed UDM objects to OX App Suite:

1. The *Provisioning Service* detects a change in the *Identity Store and Directory Service* and sends a message with the UDM object through the *Provisioning API*.
2. In the container, the *OX Connector Provisioning Consumer* receives the message and stores it as a task in its SQLite database.
3. The *OX Connector Provisioning Consumer* processes tasks in a fixed module order and sends the data for each task to the *SOAP API* in OX App Suite.
4. After the *SOAP API* successfully processes the data, the *OX Connector Provisioning Consumer* stores the object state in its database of old entries. For more information, see *Database of stored object state* (page 48).
5. After a connection error, the connector stops processing tasks and retries later. For other errors, it moves the task to the morgue.

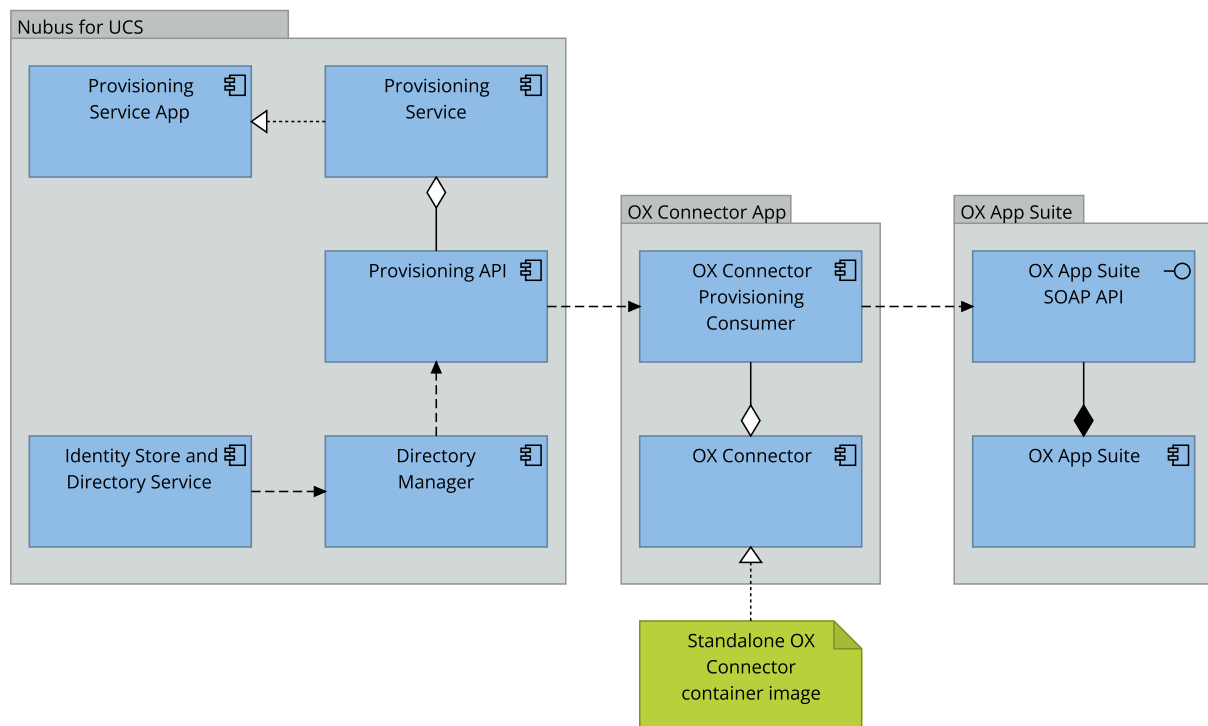


Fig. 6.3: Provisioning procedure

6.2.3 Provisioned attributes

For information about configuring the user attribute mapping, see *User attribute mapping* (page 22).

For the related group, context, and resource provisioning implementations, see the following files:

- `univention-ox-provisioning/univention/ox/provisioning/groups.py`

- `univention-ox-provisioning/univention/ox/provisioning/contexts.py`
- `univention-ox-provisioning/univention/ox/provisioning/resources.py`

6.2.4 Database of stored object state

Added in version 3.0.0.

The OX Connector stores its database file at `/var/lib/univention-appcenter/apps/ox-connector/data/ox-connector.db`. The database contains a table named `old`. Don't modify the database file directly.

Listing 6.1 shows how to inspect pending tasks on the Nubus for UCS system. The command displays a summary of the pending tasks.

For more information about provisioning tasks, see [Manage provisioning tasks](#) (page 55).

Listing 6.1: Inspect pending OX Connector tasks.

```
$ univention-app shell ox-connector \
python3 -m univention.ox.provisioning.db summarize-tasks
```

6.3 For Nubus for Kubernetes

The **OX Connector** provisions selected identity data from Nubus for Kubernetes into *OX App Suite*. It receives directory changes through the *Provisioning API*. It sends create, update, and delete requests to the *OX App Suite SOAP API*.

Before you continue reading, ensure you know [Shared architecture](#) (page 43).

6.3.1 Overview

The **OX Connector** runs as an *OX Connector Provisioning Consumer* in the Kubernetes cluster. The consumer receives changes from the *Provisioning Service* for selected Univention Directory Manager (UDM) modules. It processes the changes and provisions the corresponding *OX App Suite* objects. Fig. 6.4 shows the following components, their boundaries, and dependencies:

1. *Identity Store and Directory Service*
2. *Directory Manager*
3. *Provisioning Service*
4. *Provisioning API*
5. *OX Connector Provisioning Consumer*
6. *OX App Suite*

Subscription

A *subscription* defines the *Provisioning Service* topics for which the *OX Connector Provisioning Consumer* receives messages.

PostgreSQL database: OX Connector

The *OX Connector Provisioning Consumer* requires a PostgreSQL database for the task queue and object state. The database helps the connector keep track of objects when synchronization errors occur. The database isn't part of the *OX Connector* deployment.

For descriptions of shared components, see [Shared architecture](#) (page 43).

6.3.2 Deployment

You deploy the *OX Connector* as a Helm release in the Kubernetes cluster. The release creates a StatefulSet with the following containers:

- A main OX Connector container

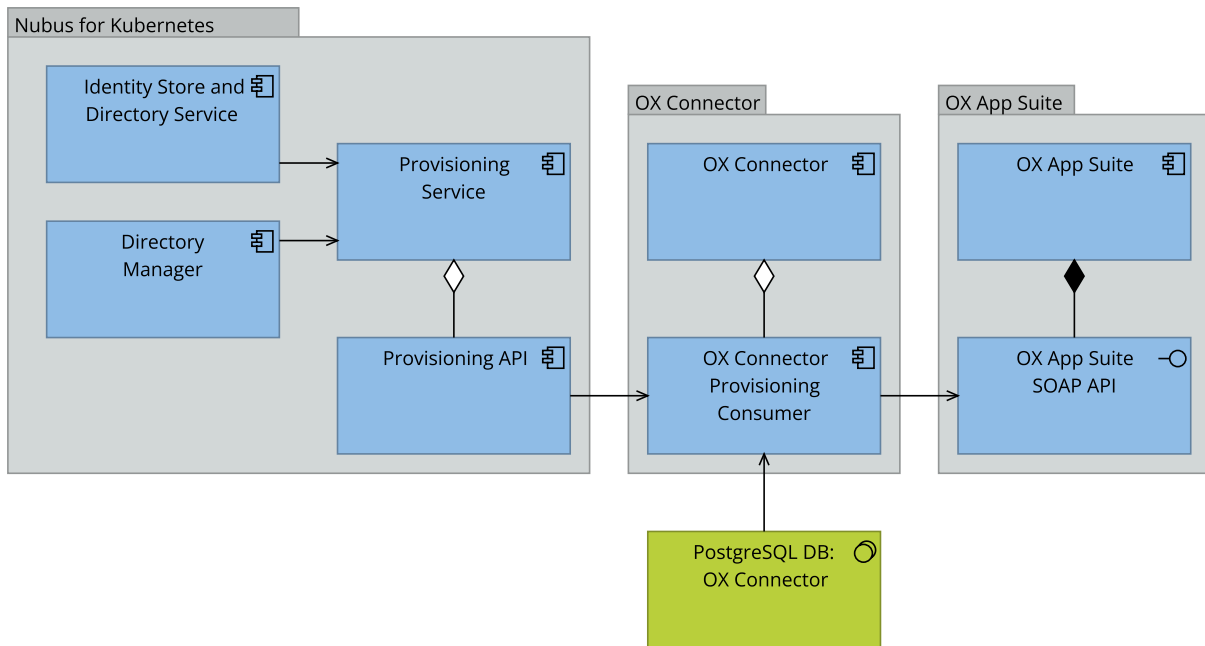


Fig. 6.4: OX Connector components for Nubus for Kubernetes

- A container that waits for the *Provisioning API*
- An initialization container that prepares the connector database

The consumer uses a configured PostgreSQL database for its task queue and object state. Fig. 6.5 shows the deployment of the *OX Connector* components.

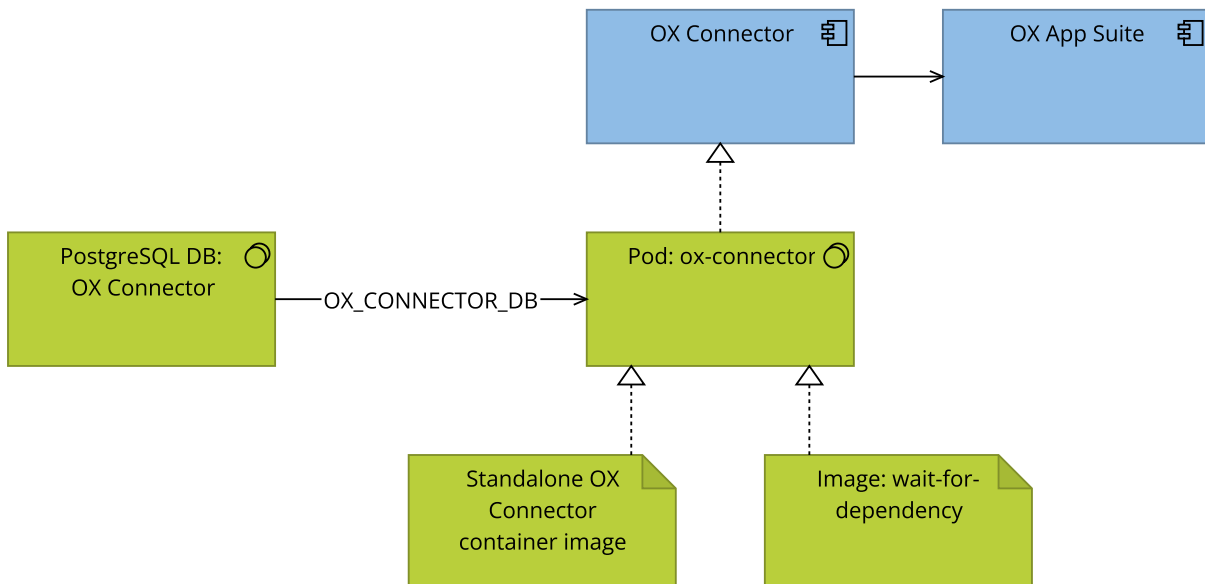


Fig. 6.5: Deployment view for *OX Connector* components

6.3.3 OX Connector Provisioning Consumer behavior

The *OX Connector* connects to the *Provisioning API* within the cluster. Nubus for Kubernetes doesn't expose the *Provisioning API* outside the cluster.

The *OX Connector Provisioning Consumer* processes one provisioning message at a time. Fig. 6.6 shows the main processing flow.

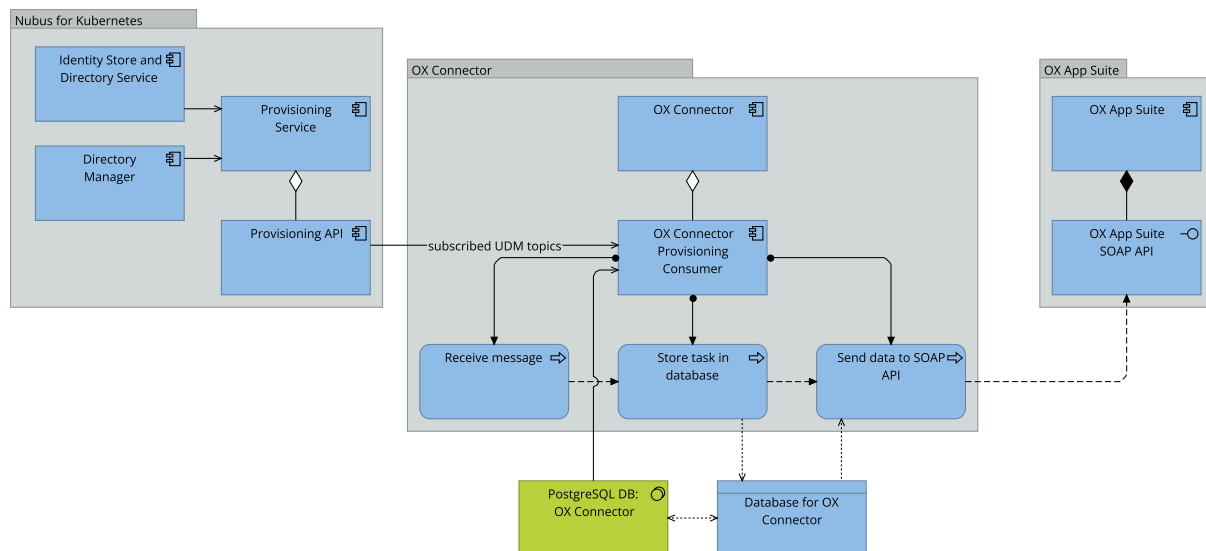


Fig. 6.6: OX Consumer provisioning flow

The following list describes the provisioning flow:

1. A change to a subscribed UDM object creates an event in the *Provisioning Service*.
2. The *Provisioning API* sends the event to the *OX Consumer* subscription.
3. The *OX Consumer* stores the event as a task in its database.
4. The consumer processes the event and sends its data to the *SOAP API* in *OX App Suite*.
5. After successful processing, the consumer stores the object state in its database.

When the message handler returns successfully, the *Provisioning API* acknowledges the message. If an exception escapes the message handler, the *Provisioning API* doesn't acknowledge the message and redelivers it. The consumer process then stops. Kubernetes restarts the process with fresh network connections.

The consumer processes context changes before dependent object types. It then processes the following object types:

- Context changes
- Access profiles
- Users
- Groups
- Functional accounts
- Shared account permissions
- Shared accounts
- Resources
- Context deletions

Detailed OX Connector Provisioning Consumer processing, click to open.

Fig. 6.7 shows the task queue, the database for *OX Connector*, and the SOAP provisioning step. The diagram also shows the relation between the *Provisioning API*, the *OX Consumer*, and *OX App Suite*.

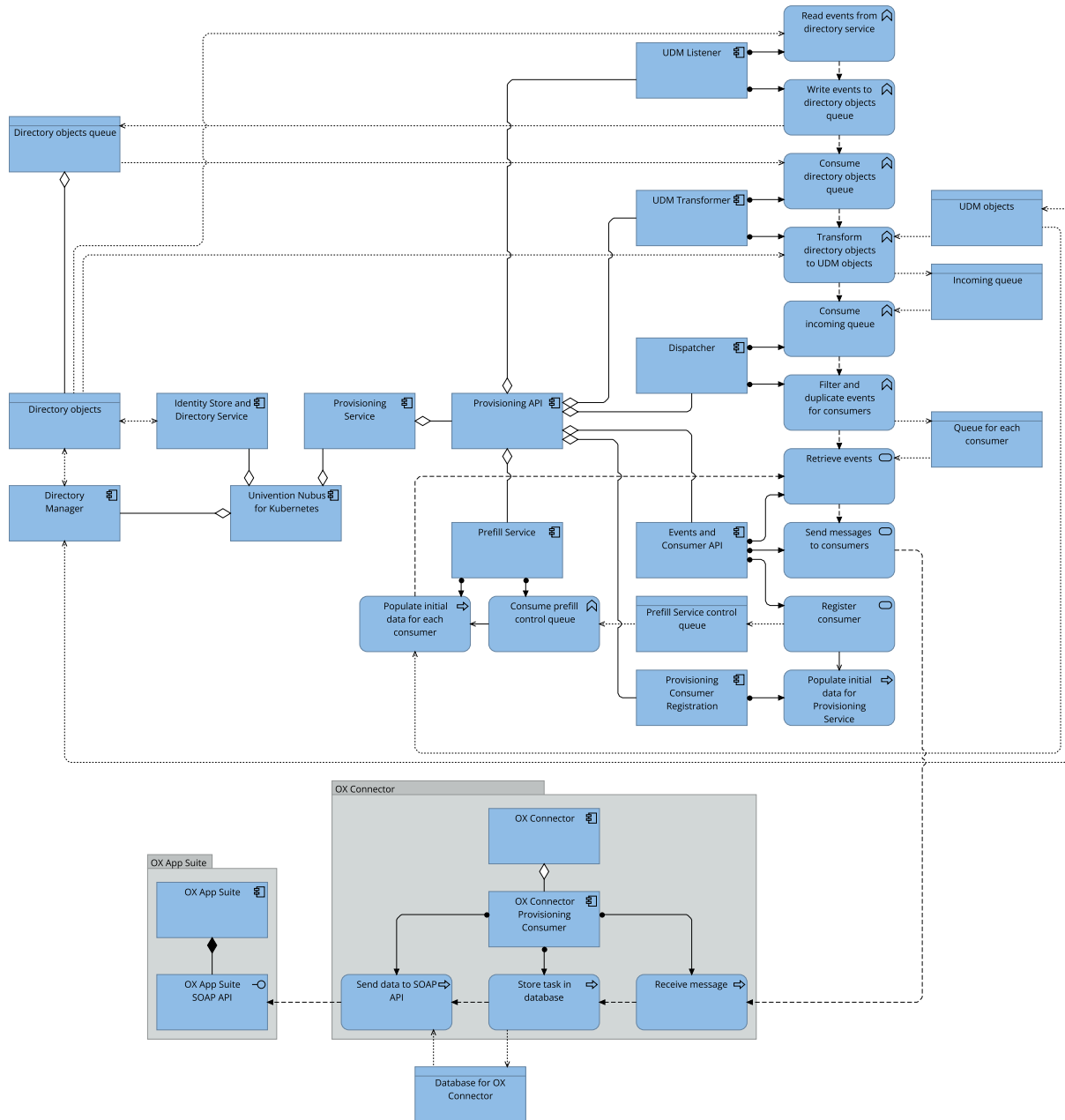


Fig. 6.7: Detailed OX Consumer provisioning flow
Click the diagram to zoom in.

6.3.4 Provisioned attributes

For information about provisioned attributes, see *Shared architecture* (page 43).

6.3.5 Database of stored object state

The *OX Connector Provisioning Consumer* stores pending tasks and object state in the configured SQL database. An initialization container prepares the database when the *OX Connector* starts.

When the `old` and `tasks` tables are empty, the initialization container requests a *prefill* from the *Provisioning Service*. It skips the request if its configuration doesn't enable resynchronization or if it can't access the required credentials.

For information about stored object state and OX internal IDs, see *Shared architecture* (page 43).

LIMITATIONS

To use the *OX Connector* reliably, read the following limitations. Each section identifies the deployments it applies to.

7.1 OX Connector integration with the OX App Suite app

Deployment — Nubus for UCS

This section applies to the Nubus for UCS deployment.

Added in version 2.1.2.

Starting with version 2.1.2 of the OX Connector app, you can use the *OX Connector* with the *OX App Suite* app from Univention App Center. The OX Connector handles provisioning, while OX App Suite provides groupware.

For OX App Suite administrator credentials on separate UCS systems, see *OX App Suite administrator* (page 3).

7.2 How OX Connector handles faulty items

Deployment — All: Nubus for UCS and Nubus for Kubernetes

This section applies to both deployments.

The *OX Connector* stores provisioning tasks in its database. It uses `OX_CONNECTOR_STOP_ON_ERROR` (page 21) to select how it handles ordinary errors during provisioning.

Nubus for UCS

In the Nubus for UCS deployment, the value defaults to `true`.

Nubus for Kubernetes

In the Nubus for Kubernetes deployment, the value defaults to `false`.

7.2.1 OX Connector continues after faulty items

If `OX_CONNECTOR_STOP_ON_ERROR` (page 21) is `false` and the *OX Connector* can't process a faulty queue item, it moves the task to the error list and continues with the remaining queue items. The connector logs the problem.

The *OX Connector* app provides a command-line interface (CLI) to manage the error list. For more information, see *Log files* (page 63) and *Manage provisioning tasks* (page 55).

Regardless of the setting, the *OX Connector* handles HTTP, connection, timeout, and OX context errors differently. It retains the task in the task list, increments its error count, and retries the task after a delay.

Nubus for UCS

When you set `OX_CONNECTOR_STOP_ON_ERROR` (page 21) to `false`, you must monitor the list of errors manually and decide whether to delete or retry a task. Meanwhile, the *OX Connector* continues to process data it receives from the *Provisioning Service*.

Nubus for Kubernetes

In the Nubus for Kubernetes deployment, the OX Connector moves tasks that fail with an ordinary error to the error list by default. The OX Connector then continues with the following tasks.

Network errors and errors while processing OX context objects remain in the task list. The consumer retries these tasks while outstanding tasks remain.

7.2.2 OX Connector stops at faulty items

If `OX_CONNECTOR_STOP_ON_ERROR` (page 21) is `true` and the *OX Connector* can't process a faulty queue item, it retains the task in the task list. The connector logs the problematic task in the *OX Connector Provisioning Consumer log file* (page 63). For more information, see *Log files* (page 63).

The *Provisioning Service* continues to add items to the queue.

Nubus for UCS

After an administrator removes the faulty queue item, the *OX Connector Provisioning Consumer* resumes processing the queue. It also processes items that the *Provisioning Service* adds.

As administrator, you need to resolve that conflict manually when it happens, see *Resolve blocked provisioning* (page 59). After the conflict resolution, the connector continues to process the provisioning queue.

Nubus for Kubernetes

In the Nubus for Kubernetes deployment, when you set `OX_CONNECTOR_STOP_ON_ERROR` (page 21) to `true`, the OX Connector retains a task that fails with an ordinary error in the task list.

7.3 No validation of access profile rights

Deployment — All: Nubus for UCS and Nubus for Kubernetes

This section applies to both deployments.

When an access profile changes, the *OX Connector* doesn't check its rights against *OX App Suite*. It writes the recognized rights to the module-access definition file and applies them to users during provisioning. If a user references an unknown access profile, the connector leaves the user's existing module-access rights unchanged.

For more information, see *OX App Suite Permission Level*⁴¹.

⁴¹ https://oxpedia.org/wiki/index.php?title=AppSuite:Permission_Level

TROUBLESHOOTING

Use this chapter to diagnose and resolve OX Connector provisioning problems. It describes the available troubleshooting resources.

Manage provisioning tasks (page 55)

Check provisioning health, handle failed or blocked tasks, and re-provision data.

Troubleshoot OX Connector on Nubus for UCS (page 63)

Inspect log files and resolve deployment-specific problems, including stale OX database IDs, duplicate display names, missing group members, and invalid identifiers.

Troubleshoot OX Connector on Nubus for Kubernetes (page 68)

Read OX Connector and Provisioning Service logs, and resolve deployment-specific provisioning problems.

Troubleshoot migration from functional accounts to shared accounts (page 73)

Recover shared accounts after an interrupted migration from functional accounts.

Collect information for a support ticket (page 74)

Gather log messages, reproduction steps, expected behavior, and affected provisioning data for Univention Support.

8.1 Manage provisioning tasks

Use the *OX Connector* command-line interface to query and manage its database. The database tracks current tasks, objects that the *OX Connector* has synchronized, and errors.

Nubus for UCS

On Nubus for UCS with the *OX Connector* app installed, you find the command-line interface at the location shown in [Listing 8.1](#).

Listing 8.1: List all commands of the CLI

```
$ /usr/sbin/univention-ox-connector-task-management --help
```

The tool operates on the **SQLite** database `/var/lib/univention-appcenter/apps/ox-connector/data/ox-connector.db`. The terminology of the tool is as follows:

Nubus for Kubernetes

To run the `univention-ox-connector-task-management` command, you need to use `kubectl` with the *execute* action on the *OX Connector* pod.

Replace the placeholder `<NAMESPACE>` with the Kubernetes namespace for the *OX Consumer*.

Use the commands in [Listing 8.2](#). The example has the pod name `ox-connector-0`.

Listing 8.2: Open a shell to the *OX Connector*

```
$ export NAMESPACE_FOR_CONSUMER="<NAMESPACE>"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get pods | grep "ox-connector"
ox-connector-0      1/1    Running    0    19h

$ export POD_NAME_FOR_OX_CONNECTOR="ox-connector-0"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management --help
```

Tasks

A database table managed by the *OX Connector*. A row represents an active task. The *OX Connector* iterates over all tasks and synchronizes them to the *OX App Suite*.

Old

A database table managed by the *OX Connector*. A row represents the state of an item when the *OX Connector* successfully synchronized it. The row is a copy of a previous task. The *OX Connector* needs it when synchronizing items that reference other items, for example, groups that contain users. The *OX Connector* also stores the database ID assigned by OX so that it can look up objects faster.

Morgue

A database table managed by the *OX Connector*. A row represents a failed task. The *OX Connector* or an administrator moved the task to this table, so the *OX Connector* doesn't process it actively. Administrators can examine items in the morgue and decide how to proceed.

8.1.1 Check provisioning health

To check the provisioning health, use the following steps:

1. Inspect the log output of the *OX Connector Provisioning Consumer*.

Nubus for UCS

To check the provisioning health, first inspect the *log output of the OX Connector Provisioning Consumer* (page 63) for warnings and errors. For more information, see *Log files* (page 63).

Nubus for Kubernetes

To check the provisioning health, first inspect the *log output of the OX Connector Provisioning Consumer* (page 68) for warnings and errors. For more information, see *Log files* (page 68).

2. Inspect the provisioning queue. The following listings show the commands. If the number of pending tasks keeps growing after a change in the LDAP directory, the *OX Connector Provisioning Consumer* or the *Provisioning Service* can't process tasks.

Nubus for UCS

Listing 8.3: Show pending tasks

```
$ /usr/sbin/univention-ox-connector-task-management summarize-tasks
$ /usr/sbin/univention-ox-connector-task-management search-tasks
```

Nubus for Kubernetes

To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.4: Show pending tasks

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management summarize-tasks

$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management search-tasks
```

3. Then inspect failed tasks in the morgue. This is relevant only if you configured the connector to *OX Connector continues after faulty items* (page 53).

Nubus for UCS

Listing 8.5: Show failed tasks in the morgue

```
$ /usr/sbin/univention-ox-connector-task-management search-morgue
```

Nubus for Kubernetes

To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.6: Show failed tasks in the morgue

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management search-morgue
```

8.1.2 Handle failed tasks

Decide how to handle failed tasks in the morgue.

Nubus for UCS

Use [Listing 8.5](#) to find the `UniventionObjectIdentifier` of the affected object. Replace `OBJECT_ID` in the following commands with this identifier.

Nubus for Kubernetes

Use [Listing 8.6](#) to find the `UniventionObjectIdentifier` of the affected object. Replace `OBJECT_ID` in the following commands with this identifier.

1. **Remove the task from the morgue:** The *OX Connector* treats the object as though it had never received the task. If the underlying object changes in the LDAP directory, the *OX Connector* can synchronize it again and create a new task.

Nubus for UCS

Run the command in [Listing 8.7](#).

Listing 8.7: Remove an item from the morgue

```
$ /usr/sbin/univention-ox-connector-task-management \
  remove-from-morgue \
  --obj-id=OBJECT_ID
```

Nubus for Kubernetes

Run the command in [Listing 8.8](#). To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.8: Remove an item from the morgue

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management \
  remove-from-morgue \
  --obj-id=OBJECT_ID
```

2. **Retry the same task:** After you resolve the problem, the *OX Connector* copies the failed task back to the task list. For example, you might first deactivate a validation rule in *OX App Suite*.

Nubus for UCS

Run the command in [Listing 8.9](#).

Listing 8.9: Retry an item from the morgue

```
$ /usr/sbin/univention-ox-connector-task-management \
  retry-from-morgue \
  --obj-id=OBJECT_ID
```

Nubus for Kubernetes

Run the command in [Listing 8.10](#). To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.10: Retry an item from the morgue

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
```

(continues on next page)

(continued from previous page)

```
--stdin \
--tty \
"$POD_NAME_FOR_OX_CONNECTOR" \
-- \
univention-ox-connector-task-management \
retry-from-morgue \
--obj-id=OBJECT_ID
```

3. **Resynchronize the object:** The *OX Connector* adds the object to the task list with its current attributes instead of the attributes from the failed synchronization. It fetches the object again from the LDAP directory. This works only for the first matching object, so asterisks might not produce the expected result.

Nubus for UCS

Run the command in [Listing 8.11](#).

Listing 8.11: Re-sync an existing item through UDM

```
$ /usr/sbin/univention-ox-connector-task-management \
resync-item \
--obj-id=OBJECT_ID
```

Nubus for Kubernetes

Run the command in [Listing 8.12](#). To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.12: Re-sync an existing item through UDM

```
$ kubectl \
--namespace "$NAMESPACE_FOR_CONSUMER" \
execute \
--stdin \
--tty \
"$POD_NAME_FOR_OX_CONNECTOR" \
-- \
univention-ox-connector-task-management \
resync-item \
--obj-id=OBJECT_ID
```

8.1.3 Resolve blocked provisioning

If provisioning has stopped, a previous change in Univention Directory Manager (UDM) might be the cause. The *OX Connector* can't process the change and retries the action until an administrator resolves the cause.

Nubus for UCS

First, see [Log files](#) (page 63). Then look for warnings and errors. If the problem isn't temporary, such as a network connectivity problem, resolve it manually.

As a last resort, move the task to the morgue. Find the task ID in the log file. It follows `tasks:` in an entry such as `uid=...; OBJECT_IDENTIFIER; tasks:TASK_ID`. Replace `TASK_ID` in the command in [Listing 8.13](#).

Listing 8.13: Move a task to the morgue

```
$ /usr/sbin/univention-ox-connector-task-management \
move-task-to-morgue \
```

(continues on next page)

(continued from previous page)

```
--task-id=TASK_ID \
--error-msg="Manual intervention after careful consideration"
```

Nubus for Kubernetes

First, see [Log files](#) (page 68). Then look for warnings and errors. If the problem isn't temporary, such as a network connectivity problem, resolve it manually.

As a last resort, move the task to the morgue. Find the task ID in the log file. It follows `tasks:` in an entry such as `uid=...; OBJECT_IDENTIFIER; tasks:TASK_ID`. Replace `TASK_ID` in the command in [Listing 8.14](#).

To set the proper environment variables in the following listing, use the commands in [Listing 8.2](#).

Listing 8.14: Move a task to the morgue

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management \
  move-task-to-morgue \
  --task-id=TASK_ID \
  --error-msg="Manual intervention after careful consideration"
```

8.1.4 Re-provision all data

To re-provision all data, recreate the *OX Connector* subscription and enable *prefill*, as described below. The *Provisioning Service* sends existing Univention Directory Manager (UDM) objects from subscribed modules to the *OX Connector*. The *OX Connector Provisioning Consumer* adds them to its queue.

Warning

Depending on the number of users and groups in the Nubus for UCS LDAP directory, this task can take a long time.

Avoid re-provisioning all data.

The Provisioning Service doesn't add deleted UDM objects to the queue. Therefore, the *OX Connector* doesn't run delete operations during re-provisioning. Previously deleted UDM objects aren't removed from OX during this procedure.

Nubus for UCS

To re-provision all data, run the commands in [Listing 8.15](#) on the [Primary Directory Node](#)⁴². The commands do the following:

1. Set up configuration parameters, such as base URL, administrator password, and subscription password.
2. Delete the existing subscription.
3. Configure the connector subscription to subscribe to all relevant UDM modules.
4. Create the subscription using the configuration from the JSON file.
5. Delete the subscription configuration.

⁴² <https://docs.software-univention.de/ucs-operation/5.2/en/glossary.html#term-Primary-Directory-Node>

6. Save the *Provisioning Service* credentials to a file in the *OX Connector* configuration directory and restrict the file permissions.
7. Restart the *OX Connector* app.

Listing 8.15: Re-provisioning all UDM objects to OX App Suite

```
$ export BASE_URL="https://$(ucr get ldap/master)/univention/provisioning"
$ export ADMIN_PASSWORD="$(python3 -c 'import json; print(json.load(open("/etc/
→provisioning-secrets.json"))["PROVISIONING_API_ADMIN_PASSWORD"])' )"
$ export SUBSCRIPTION_PASSWORD="$(openssl rand -hex 32)"
$ curl --user "admin:$ADMIN_PASSWORD" \
  -X DELETE "$BASE_URL/v1/subscriptions/ox-connector" || true
$ umask 077
$ cat > /tmp/ox-connector-subscription.json <<EOF
{
  "name": "ox-connector",
  "realms_topics": [
    {"realm": "udm", "topic": "users/user"},
    {"realm": "udm", "topic": "groups/group"},
    {"realm": "udm", "topic": "oxmail/oxcontext"},
    {"realm": "udm", "topic": "oxmail/accessprofile"},
    {"realm": "udm", "topic": "oxresources/oxresources"},
    {"realm": "udm", "topic": "oxmail/functional_account"},
    {"realm": "udm", "topic": "oxmail/shared_account"},
    {"realm": "udm", "topic": "oxmail/shared_account_permission"}
  ],
  "request_prefill": true,
  "password": "$SUBSCRIPTION_PASSWORD"
}
EOF
$ curl --fail --user "admin:$ADMIN_PASSWORD" \
  -H "Content-Type: application/json" \
  -X POST "$BASE_URL/v1/subscriptions" \
  --data @/tmp/ox-connector-subscription.json \
  || { rm -f /tmp/ox-connector-subscription.json; exit 1; }
$ rm -f /tmp/ox-connector-subscription.json
$ printf 'export PROVISIONING_API_USERNAME=ox-connector\nexport PROVISIONING_API_
→PASSWORD=%s\n' \
  "$SUBSCRIPTION_PASSWORD" \
  > /var/lib/univention-appcenter/apps/ox-connector/conf/provisioning.env
$ chmod 640 /var/lib/univention-appcenter/apps/ox-connector/conf/provisioning.env
$ univention-app restart ox-connector
```

Nubus for Kubernetes

Before you can run the commands in Listing 8.18 you need to complete the following prerequisites:

1. Open an access to the *Provisioning API* endpoint. See [Access to Provisioning API endpoint⁴³](#) in *Univention Nubus - Customization and Modification Manual* [5].
2. Define the following environment variables:
 - NAMESPACE_FOR_CONSUMER, see Listing 8.16. Replace the placeholder <NAMESPACE> with the Kubernetes namespace for the *OX Consumer*.
 - POD_NAME_FOR_OX_CONNECTOR, see Listing 8.16.
 - PROVISIONING_API_ADMIN_SECRET_NAME, see Listing 8.17

⁴³ <https://docs.software-univention.de/nubus-customization/1.x/en/api/provisioning.html#customization-api-provisioning-endpoint-access>

Listing 8.16: Define POD_NAME_FOR_OX_CONNECTOR

```
$ export NAMESPACE_FOR_CONSUMER="<NAMESPACE>"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get pods | grep "ox-connector"
ox-connector-0      1/1      Running    0    19h

$ export POD_NAME_FOR_OX_CONNECTOR="ox-connector-0"
```

Listing 8.17: Define PROVISIONING_API_ADMIN_SECRET_NAME

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get secret | grep provisioning-api-admin
ums-provisioning-api-admin    Opaque    1    73d

$ export PROVISIONING_API_ADMIN_SECRET_NAME="ums-provisioning-api-admin"
```

To re-provision all data, run the commands in Listing 8.18. The commands do the following:

1. Set up configuration parameters, such as base URL, administrator password, and subscription password.
2. Delete the existing subscription.
3. Configure the connector subscription to subscribe to all relevant UDM modules.
4. Create the subscription using the configuration from the JSON file.
5. Delete the subscription configuration.
6. Save the *Provisioning Service* credentials to a file in the directory where you run the commands and restrict the file permissions.
7. Recreate the pod for the *OX Connector*.

Listing 8.18: Re-provisioning all UDM objects to OX App Suite

```
$ export BASE_URL="http://localhost:7777"
$ export ADMIN_PASSWORD="$(kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get secret \
$PROVISIONING_API_ADMIN_SECRET_NAME \
  -o json \
  | jq -r ".data.password" \
  | base64 -d)"
$ export SUBSCRIPTION_PASSWORD="$(openssl rand -hex 32)"
$ curl --user "admin:$ADMIN_PASSWORD" \
  -X DELETE "$BASE_URL/v1/subscriptions/ox-connector" || true
$ umask 077
$ cat > /tmp/ox-connector-subscription.json <<EOF
{
  "name": "ox-connector",
  "realms_topics": [
    {"realm": "udm", "topic": "users/user"},
    {"realm": "udm", "topic": "groups/group"},
    {"realm": "udm", "topic": "oxmail/oxcontext"},
    {"realm": "udm", "topic": "oxmail/accessprofile"},
    {"realm": "udm", "topic": "oxresources/oxresources"},
    {"realm": "udm", "topic": "oxmail/functional_account"}
  ]
}
```

(continues on next page)

(continued from previous page)

```

    {"realm":"udm", "topic":"oxmail/shared_account"},
    {"realm":"udm", "topic":"oxmail/shared_account_permission"}
  ],
  "request_prefill": true,
  "password": "$SUBSCRIPTION_PASSWORD"
}
EOF
$ curl --fail --user "admin:$ADMIN_PASSWORD" \
  -H "Content-Type: application/json" \
  -X POST "$BASE_URL/v1/subscriptions" \
  --data @/tmp/ox-connector-subscription.json \
  || { rm -f /tmp/ox-connector-subscription.json; exit 1; }
$ rm -f /tmp/ox-connector-subscription.json
$ printf 'export PROVISIONING_API_USERNAME=ox-connector\nexport PROVISIONING_
↪API_PASSWORD=%s\n' \
  "$SUBSCRIPTION_PASSWORD" \
  > provisioning.env
$ chmod 640 provisioning.env
$ kubectl delete pod "$POD_NAME_FOR_OX_CONNECTOR"

```

Caution

The *OX Connector* can delete objects based on the data that it receives. For example, it deletes a group object when `isOxGroup = False`.

8.2 Troubleshoot OX Connector on Nubus for UCS

Use this page to troubleshoot problems with the *OX Connector* app on Nubus for UCS. It explains how to inspect log files. It also covers known problems, including duplicate display names, missing group members, and invalid identifier values.

8.2.1 Log files

The *OX Connector* app writes logs to several locations. This section lists the log file per involved component.

OX Connector Provisioning Consumer

Standard output of the *OX Connector* container. To read the output, run the command in [Listing 8.19](#).

Contains log information from the *OX Connector Provisioning Consumer* about object create, update, and delete actions. It also reports warnings and errors when the *OX Connector* configuration isn't correct or the connector can't connect to the *SOAP API*.

Listing 8.19: View the log output of the OX Connector Provisioning Consumer

```
$ univention-app logs ox-connector
```

Provisioning Service

`/var/log/univention/listener_modules/nubus-provisioning.log`

Contains log information from the *Provisioning Service* about changes detected in the LDAP directory and delivered to subscribed services. The Provisioning Service containers write additional log information to `/var/log/syslog`.

Database management script

`/var/lib/univention-appcenter/apps/ox-connector/data/univention-ox-connector-task-management.log`

Contains log information from the database management script described in [Manage provisioning tasks](#) (page 55).

App Center

`/var/log/univention/appcenter.log`

Contains log information about *App Center* activities.

The *App Center* writes *OX Connector*-related information to this file when you run app lifecycle tasks, such as installing, updating, or uninstalling the app, or when you change the app settings.

For information about *App Center* logging and diagnostics, see [Logging and diagnostics](#)⁴⁴ in *Nubus for UCS 5.2 - Operation Manual* [1].

Domain join

`/var/log/univention/join.log`

Contains log information from domain join processes. When the *App Center* installs *OX Connector*, the app also joins the domain.

For information about the domain join process, see [Domain join process](#)⁴⁵ in *Nubus for UCS 5.2 - Operation Manual* [1].

8.2.2 Ensure OX database ID integrity

The *OX Connector* caches the internal IDs that *OX App Suite* assigns to objects. After you restore an *OX App Suite* database backup, the cached IDs can become stale or inconsistent. For more information about the cache, see [Database of stored object state](#) (page 48).

To rebuild the cache, run the command in [Listing 8.20](#).

Listing 8.20: Rebuild cache for *internal ID*

```
$ /usr/sbin/univention-ox-connector-task-management rewrite-ox-db-id
```

Tip

Retrieve all users per context in one request

Rebuilding the cache can take a long time and depends on the number of users in the *OX App Suite* database.

The command in [Listing 8.21](#) can speed up cache rebuilding because it retrieves up to 1,000 users from one context per request.

Listing 8.21: Increase performance on rebuilding the cache

```
$ /usr/sbin/univention-ox-connector-task-management \
  rewrite-ox-db-id \
  --build-cache-size=1000
```

Warning

Memory consumption

On the Nubus for UCS system with *OX Connector*, the rebuild process can use up to 1 GB of memory per 10,000 users in the *OX App Suite* database.

System load

Furthermore, the process may generate a lot of load on the *OX App Suite* system and the *OX Connector* app.

⁴⁴ <https://docs.software-univention.de/ucs-operation/5.2/en/lifecycle/app-center.html#lifecycle-app-center-troubleshooting-logging>

⁴⁵ <https://docs.software-univention.de/ucs-operation/5.2/en/domain-infrastructure/domain-join.html#domain-infrastructure-join-process>

8.2.3 Duplicate display names

Since *OX Connector* version 2.2.0, the UDM property *oxDisplayName* no longer has a unique constraint.

If *OX App Suite* isn't configured to allow duplicate display names, *SOAP API* calls fail with the exception as shown in [Listing 8.22](#).

Listing 8.22: Exception when OX App Suite doesn't allow duplicate display names

```
2023-05-30 11:59:31 WARNING Traceback (most recent call last):
2023-05-30 11:59:31 WARNING   File "/tmp/univention-ox-connector.listener_trigger",
↳ line 324, in run_on_files
2023-05-30 11:59:31 WARNING       f(obj)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ provisioning/__init__.py", line 86, in run
2023-05-30 11:59:31 WARNING       modify_user(obj)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ provisioning/users.py", line 420, in modify_user
2023-05-30 11:59:31 WARNING       user.modify()
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ soap/backend.py", line 477, in modify
2023-05-30 11:59:31 WARNING       super(SoapUser, self).modify()
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ soap/backend.py", line 180, in modify
2023-05-30 11:59:31 WARNING       self.service(self.context_id).change(obj)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ soap/services.py", line 536, in change
2023-05-30 11:59:31 WARNING       return self._call_ox('change', usrddata=user)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/univention/ox/
↳ soap/services.py", line 163, in _call_ox
2023-05-30 11:59:31 WARNING       return getattr(service, func)(**kwargs)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/zeep/proxy.py
↳ ", line 46, in __call__
2023-05-30 11:59:31 WARNING       return self._proxy._binding.send(
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsd/
↳ bindings/soap.py", line 135, in send
2023-05-30 11:59:31 WARNING       return self.process_reply(client, operation_obj,
↳ response)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsd/
↳ bindings/soap.py", line 229, in process_reply
2023-05-30 11:59:31 WARNING       return self.process_error(doc, operation)
2023-05-30 11:59:31 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsd/
↳ bindings/soap.py", line 329, in process_error
2023-05-30 11:59:31 WARNING       raise Fault(
2023-05-30 11:59:31 WARNING   zeep.exceptions.Fault: The displayname is already used;
↳ exceptionId 1170523631-4
```

To allow duplicate display names, add the properties in [Listing 8.23](#) to the `user.properties` file in *OX App Suite*.

Listing 8.23: Configuration to allow duplicate display names in OX App Suite

```
com.openexchange.user.enforceUniqueDisplayName=false
com.openexchange.folderstorage.database.preferDisplayName=false
```

Note

The *OX App Suite* installation from the *App Center* configures these properties by default.

8.2.4 Missing group members

When the *OX Connector* synchronizes a group, it needs the *internal ID* of every group member. For more information, see *Database of stored object state* (page 48). The connector looks up each member in its database of old entries. If a user belongs to a group but isn't in that database, the *OX Connector* skips the user without failing. It logs a message, as shown in *Listing 8.24*.

Re-provision the missing user object manually. For example, re-provision `uid=oxuser1,cn=users,dc=example,dc=com`. Follow *Handle failed tasks* (page 57) to synchronize the missing user. When the *OX Connector* next processes the group object, the *OX Connector Provisioning Consumer* adds the user to the group again.

Listing 8.24: Log message for missing group members

```
2024-11-15 16:06:33 INFO      Group will be OX Group
2024-11-15 16:06:33 INFO      Group wants user as member. But the user is unknown.↵
↵Ignoring...
```

8.2.5 Invalid values for OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER

A UDM user property, or a UDM group property for *OX_GROUP_IDENTIFIER* (page 20), is valid only when it contains a single value that isn't `None`. If the configured UDM property contains an empty value or a list of values, the *OX Connector* enters an error state. Set a valid value to resolve the error.

Setting an invalid value for the app settings *OX_USER_IDENTIFIER* (page 19) or *OX_GROUP_IDENTIFIER* (page 20) leads to the errors in *Listing 8.25* or *Listing 8.26*.

Listing 8.25: Error caused by an invalid value for app settings
OX_USER_IDENTIFIER

```
2024-01-11 13:57:39 WARNING Traceback (most recent call last):
2024-01-11 13:57:39 WARNING   File "/tmp/univention-ox-connector.listener_trigger
↵", line 351, in run_on_files
2024-01-11 13:57:39 WARNING       function(obj)
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↵ox/provisioning/__init__.py", line 86, in run
2024-01-11 13:57:39 WARNING       modify_user(obj)
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↵ox/provisioning/users.py", line 454, in modify_user
2024-01-11 13:57:39 WARNING       user.modify()
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↵ox/soap/backend.py", line 475, in modify
2024-01-11 13:57:39 WARNING       super(SoapUser, self).modify()
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↵ox/soap/backend.py", line 176, in modify
2024-01-11 13:57:39 WARNING       assert self.name is not None
2024-01-11 13:57:39 WARNING   No name for this attribute. Missing or misconfigured↵
↵identifier app settings
2024-01-11 13:57:39 WARNING   (OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER) might be↵
↵the reason, see
2024-01-11 13:57:39 WARNING   https://docs.software-univention.de/ox-connector-app/
↵latest/troubleshooting.html#invalid-values-for-ox-user-identifier-or-ox-group-
↵identifier
2024-01-11 13:57:39 WARNING   for more information.
```

Listing 8.26: Error caused by an invalid value for app settings

OX_GROUP_IDENTIFIER

```

setting "users" udm property for groups
2024-01-11 13:59:36 WARNING Traceback (most recent call last):
2024-01-11 13:59:36 WARNING   File "/tmp/univention-ox-connector.listener_trigger
↳", line 351, in run_on_files
2024-01-11 13:59:36 WARNING       function(obj)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↳ox/provisioning/__init__.py", line 108, in run
2024-01-11 13:59:36 WARNING       modify_group(new_obj)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↳ox/provisioning/groups.py", line 146, in modify_group
2024-01-11 13:59:36 WARNING       group.modify()
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↳ox/soap/backend.py", line 180, in modify
2024-01-11 13:59:36 WARNING       self.service(self.context_id).change(obj)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↳ox/soap/services.py", line 607, in change
2024-01-11 13:59:36 WARNING       return self._call_ox('change', grp=grp)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↳ox/soap/services.py", line 194, in _call_ox
2024-01-11 13:59:36 WARNING       return getattr(service, func)(**kwargs)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/proxy.py
↳", line 46, in __call__
2024-01-11 13:59:36 WARNING       return self._proxy._binding.send(
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsdli
↳bindings/soap.py", line 123, in send
2024-01-11 13:59:36 WARNING       envelope, http_headers = self._create(
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsdli
↳bindings/soap.py", line 73, in _create
2024-01-11 13:59:36 WARNING       serialized = operation_obj.create(*args, **kwargs)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsdli
↳definitions.py", line 224, in create
2024-01-11 13:59:36 WARNING       return self.input.serialize(*args, **kwargs)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/wsdli
↳messages/soap.py", line 79, in serialize
2024-01-11 13:59:36 WARNING       self.body.render(body, body_value)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/
↳elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING       self._render_value_item(parent, value, render_
↳path)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/
↳elements/element.py", line 256, in _render_value_item
2024-01-11 13:59:36 WARNING       return self.type.render(node, value, None, render_
↳path)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/
↳types/complex.py", line 307, in render
2024-01-11 13:59:36 WARNING       element.render(node, element_value, child_path)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/
↳elements/indicators.py", line 256, in render
2024-01-11 13:59:36 WARNING       element.render(parent, element_value, child_path)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/
↳elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING       self._render_value_item(parent, value, render_
↳path)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/zeep/xsd/

```

(continues on next page)

(continued from previous page)

```

↪elements/element.py", line 255, in _render_value_item
2024-01-11 13:59:36 WARNING      return value._xsd_type.render(node, value, xsd_
↪type, render_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/complex.py", line 307, in render
2024-01-11 13:59:36 WARNING      element.render(node, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/indicators.py", line 256, in render
2024-01-11 13:59:36 WARNING      element.render(parent, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING      self._render_value_item(parent, value, render_
↪path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 256, in _render_value_item
2024-01-11 13:59:36 WARNING      return self.type.render(node, value, None, render_
↪path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/simple.py", line 96, in render
2024-01-11 13:59:36 WARNING      node.text = value if isinstance(value, etree.
↪CDATA) else self.xmlvalue(value)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/builtins.py", line 27, in _wrapper
2024-01-11 13:59:36 WARNING      raise ValueError(
2024-01-11 13:59:36 WARNING      ValueError: The String type doesn't accept_
↪collections as value

```

8.3 Troubleshoot OX Connector on Nubus for Kubernetes

Use this page to troubleshoot problems with the *OX Connector* on Nubus for Kubernetes. It explains how to inspect log files. It also covers known problems, including missing group members, and invalid identifier values.

8.3.1 Log files

The *OX Connector* has the following locations for relevant logs. This section describes how to read the logs through **kubect1**. You can use other tools that give you access to the same information in the respective pods.

The following listings need the same environment variables. Add them to your shell with the commands in [Listing 8.27](#).

Listing 8.27: Prepare environment variables

```
$ export NAMESPACE_FOR_CONSUMER="<NAMESPACE>"
```

OX Connector Provisioning Consumer

The main container in the `ox-connector` pod provides the log files for the *OX Connector*. To read the logs with **kubect1**, use the following steps. You find all the commands in [Listing 8.28](#).

1. Find out the name of the *OX Connector* pod and remember it. The example has the pod name `ox-connector-0`.
2. Open the log file of the main container of the *OX Connector* pod.

Listing 8.28: Read the log files of the *OX Connector*

```
$ kubect1 \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
```

(continues on next page)

(continued from previous page)

```

get pods | grep "ox-connector"
ox-connector-0      1/1      Running    0   19h

$ export POD_NAME_FOR_OX_CONNECTOR="ox-connector-0"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  logs \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -c main

```

Provisioning Service

The *Provisioning Service* is a component in Nubus for Kubernetes. To read the logs with **kubectl**, use the following steps. You find all the commands in [Listing 8.29](#).

1. Find out the name of the *Provisioning Service* API pod and remember it. The example has the pod name `ums-provisioning-api-968d7c9b7-26brx`.
2. Open the log file of the main container of the *OX Connector* pod.

Listing 8.29: Read the log files of the *Provisioning Service*

```

$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get pods | grep "provisioning-api"
ums-provisioning-api-968d7c9b7-26brx    1/1      Running    0   20h

$ export POD_NAME_FOR_PROVISIONING_API="ums-provisioning-api-968d7c9b7-26brx"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  logs \
  "$POD_NAME_FOR_PROVISIONING_API" \
  -c main

```

See also

kubectl logs | [Kubernetes](#)⁴⁶

for more information of available parameters for **kubectl logs**. For example, to follow the logs stream, use the option `follow`.

8.3.2 Ensure OX database ID integrity

The *OX Connector* caches the internal IDs that *OX App Suite* assigns to objects. After you restore an *OX App Suite* database backup, the cached IDs can become stale or inconsistent. For more information about the cache, see [Database of stored object state](#) (page 52).

To set the proper environment variables in the following listing, use the commands in [Listing 8.27](#). To rebuild the cache, run the command in [Listing 8.30](#).

Listing 8.30: Rebuild cache for *internal ID*

```

$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  get pods | grep "ox-connector"
ox-connector-0      1/1      Running    0   19h

```

(continues on next page)

⁴⁶ https://kubernetes.io/docs/reference/kubectl/generated/kubectl_logs/

(continued from previous page)

```
$ export POD_NAME_FOR_OX_CONNECTOR="ox-connector-0"
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management \
  rewrite-ox-db-id
```

Tip**Retrieve all users per context in one request**

Rebuilding the cache can take a long time and depends on the number of users in the *OX App Suite* database.

The command in [Listing 8.31](#) can speed up cache rebuilding because it retrieves up to 1,000 users from one context per request.

Listing 8.31: Increase performance on rebuilding the cache

```
$ kubectl \
  --namespace "$NAMESPACE_FOR_CONSUMER" \
  execute \
  --stdin \
  --tty \
  "$POD_NAME_FOR_OX_CONNECTOR" \
  -- \
  univention-ox-connector-task-management \
  rewrite-ox-db-id \
  --build-cache-size=1000
```

Warning**Memory consumption**

On Nubus for Kubernetes system with the *OX Connector*, the rebuild process can use up to 1 GB of memory per 10,000 users in the *OX App Suite* database.

System load

Furthermore, the process may generate a lot of load on the *OX App Suite* system and the *OX Connector* app.

8.3.3 Missing group members

When the *OX Connector* synchronizes a group, it needs the *internal ID* of every group member. For more information, see [Database of stored object state](#) (page 52). The connector looks up each member in its database of old entries. If a user belongs to a group but isn't in that database, the *OX Connector* skips the user without failing. It logs a message, as shown in [Listing 8.32](#).

Re-provision the missing user object manually. For example, re-provision `uid=oxuser1,cn=users,dc=example,dc=com`. Follow [Handle failed tasks](#) (page 57) to synchronize the missing user. When the *OX Connector* next processes the group object, the *OX Connector Provisioning Consumer* adds the user to the group again.

Listing 8.32: Log message for missing group members

```
2024-11-15 16:06:33 INFO      Group will be OX Group
2024-11-15 16:06:33 INFO      Group wants user as member. But the user is unknown.↵
↪Ignoring...
```

8.3.4 Invalid values for OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER

A UDM user property, or a UDM group property for `OX_GROUP_IDENTIFIER` (page 20), is valid only when it contains a single value that isn't `None`. If the configured UDM property contains an empty value or a list of values, the *OX Connector* enters an error state. Set a valid value to resolve the error.

Setting an invalid value for the app settings `OX_USER_IDENTIFIER` (page 19) or `OX_GROUP_IDENTIFIER` (page 20) leads to the errors in Listing 8.33 or Listing 8.34. You find the log information in the *OX Consumer logs* (page 68).

Listing 8.33: Error caused by an invalid value for app settings

OX_USER_IDENTIFIER

```
2024-01-11 13:57:39 WARNING Traceback (most recent call last):
2024-01-11 13:57:39 WARNING   File "/tmp/univention-ox-connector.listener_trigger
↪", line 351, in run_on_files
2024-01-11 13:57:39 WARNING       function(obj)
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/provisioning/__init__.py", line 86, in run
2024-01-11 13:57:39 WARNING       modify_user(obj)
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/provisioning/users.py", line 454, in modify_user
2024-01-11 13:57:39 WARNING       user.modify()
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/soap/backend.py", line 475, in modify
2024-01-11 13:57:39 WARNING       super(SoapUser, self).modify()
2024-01-11 13:57:39 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/soap/backend.py", line 176, in modify
2024-01-11 13:57:39 WARNING       assert self.name is not None
2024-01-11 13:57:39 WARNING No name for this attribute. Missing or misconfigured.↵
↪identifier app settings
2024-01-11 13:57:39 WARNING (OX_USER_IDENTIFIER or OX_GROUP_IDENTIFIER) might be↵
↪the reason, see
2024-01-11 13:57:39 WARNING https://docs.software-univention.de/ox-connector-app/
↪latest/troubleshooting.html#invalid-values-for-ox-user-identifier-or-ox-group-
↪identifier
2024-01-11 13:57:39 WARNING for more information.
```

Listing 8.34: Error caused by an invalid value for app settings

OX_GROUP_IDENTIFIER

```
setting "users" udm property for groups
2024-01-11 13:59:36 WARNING Traceback (most recent call last):
2024-01-11 13:59:36 WARNING   File "/tmp/univention-ox-connector.listener_trigger
↪", line 351, in run_on_files
2024-01-11 13:59:36 WARNING       function(obj)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/provisioning/__init__.py", line 108, in run
2024-01-11 13:59:36 WARNING       modify_group(new_obj)
2024-01-11 13:59:36 WARNING   File "/usr/lib/python3.9/site-packages/univention/
↪ox/provisioning/groups.py", line 146, in modify_group
2024-01-11 13:59:36 WARNING       group.modify()
```

(continues on next page)

(continued from previous page)

```

2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/univention/
↪ox/soap/backend.py", line 180, in modify
2024-01-11 13:59:36 WARNING      self.service(self.context_id).change(obj)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/univention/
↪ox/soap/services.py", line 607, in change
2024-01-11 13:59:36 WARNING      return self._call_ox('change', grp=grp)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/univention/
↪ox/soap/services.py", line 194, in _call_ox
2024-01-11 13:59:36 WARNING      return getattr(service, func)(**kwargs)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/proxy.py
↪", line 46, in __call__
2024-01-11 13:59:36 WARNING      return self._proxy._binding.send(
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/wsdli/
↪bindings/soap.py", line 123, in send
2024-01-11 13:59:36 WARNING      envelope, http_headers = self._create(
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/wsdli/
↪bindings/soap.py", line 73, in _create
2024-01-11 13:59:36 WARNING      serialized = operation_obj.create(*args, **kwargs)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/wsdli/
↪definitions.py", line 224, in create
2024-01-11 13:59:36 WARNING      return self.input.serialize(*args, **kwargs)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/wsdli/
↪messages/soap.py", line 79, in serialize
2024-01-11 13:59:36 WARNING      self.body.render(body, body_value)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING      self._render_value_item(parent, value, render_
↪path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 256, in _render_value_item
2024-01-11 13:59:36 WARNING      return self.type.render(node, value, None, render_
↪path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/complex.py", line 307, in render
2024-01-11 13:59:36 WARNING      element.render(node, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/indicators.py", line 256, in render
2024-01-11 13:59:36 WARNING      element.render(parent, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING      self._render_value_item(parent, value, render_
↪path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 255, in _render_value_item
2024-01-11 13:59:36 WARNING      return value._xsd_type.render(node, value, xsd_
↪type, render_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/complex.py", line 307, in render
2024-01-11 13:59:36 WARNING      element.render(node, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/indicators.py", line 256, in render
2024-01-11 13:59:36 WARNING      element.render(parent, element_value, child_path)
2024-01-11 13:59:36 WARNING      File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 232, in render
2024-01-11 13:59:36 WARNING      self._render_value_item(parent, value, render_
↪path)

```

(continues on next page)

(continued from previous page)

```

2024-01-11 13:59:36 WARNING    File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪elements/element.py", line 256, in _render_value_item
2024-01-11 13:59:36 WARNING    return self.type.render(node, value, None, render_
↪path)
2024-01-11 13:59:36 WARNING    File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/simple.py", line 96, in render
2024-01-11 13:59:36 WARNING    node.text = value if isinstance(value, etree.
↪CDATA) else self.xmlvalue(value)
2024-01-11 13:59:36 WARNING    File "/usr/lib/python3.9/site-packages/zeep/xsd/
↪types/builtins.py", line 27, in _wrapper
2024-01-11 13:59:36 WARNING    raise ValueError(
2024-01-11 13:59:36 WARNING ValueError: The String type doesn't accept
↪collections as value

```

8.4 Troubleshoot migration from functional accounts to shared accounts

During the *migration of functional accounts to shared accounts* (page 39), a network failure or another unexpected error can leave a shared account half-configured. You might encounter one of the following states:

Functional account still exists

The functional account is still present, and the shared account is partially configured. Rerun the script with the same parameters as before to retry the migration.

Functional account no longer exists

The functional account no longer exists, so only the final migration step remains. The remaining step is to modify the email address of the shared account and remove the `tmp_` prefix. To remove the prefix, use either the *Management UI* or the `udm` command.

Management UI

Deployment — All: Nubus for UCS and Nubus for Kubernetes

This section applies to both deployments.

Use the *Management UI* on Nubus for Kubernetes or in cases where you don't have access to the UDM command-line. Use the following steps:

1. Sign in to the *Management UI* and navigate to the [LDAP directory module](#)⁴⁷.
2. Select the container for the shared accounts. The default container is `cn=shared_accounts, cn=open-xchange, <ldap_base>`.
3. Open the affected shared account.
4. Change the email address and remove the `tmp_` prefix.
5. Click *Save*.

Verify that the shared account uses the expected email address and no longer has the `tmp_` prefix.

UDM command-line

Deployment — Nubus for UCS

This section applies to the Nubus for UCS deployment.

To remove the prefix with the `udm` command in Nubus for UCS, run the command shown in [Listing 8.35](#). Define the following parameters:

⁴⁷ <https://docs.software-univention.de/nubus-manual/1.x/en/management/domain/ldap.html#nubus-domain-ldap>

SHARED_ACCOUNT

The DN of the affected shared account, for example "cn=test,cn=shared_accounts,cn=open-xchange,\$(ucr get ldap/base) "

EMAIL

The email address of the shared account.

After you ran the command, verify that the shared account uses the expected email address and no longer has the tmp_ prefix.

Listing 8.35: Remove the tmp_ prefix from the email address of a shared account

```
$ export SHARED_ACCOUNT="<DN of affected shared account>"
$ export EMAIL="<email address of the shared account>"
$ udm \
  oxmail/shared_account \
  modify \
  --dn "$SHARED_ACCOUNT" \
  --set mailPrimaryAddress="$EMAIL"
```

8.5 Collect information for a support ticket

Before you open a support ticket, collect the following information so that Univention Support can investigate the issue:

- Relevant messages and tracebacks:

Nubus for UCS

Messages and tracebacks from *Log files* (page 63), especially from the *OX Connector Provisioning Consumer logs* (page 63).

Nubus for Kubernetes

Messages and tracebacks from *Log files* (page 68), especially from the *OX Connector Provisioning Consumer logs* (page 68).

- Steps that reproduce the unexpected behavior.
- The expected behavior.
- Provisioning data that causes the error.

CHANGELOG

This page contains the changelog for the OX Connector in the deployments as *OX Connector app for Nubus for UCS* (page 75) and as *OX Connector for Nubus for Kubernetes* (page 84).

9.1 Changelog OX Connector app

This changelog documents all notable changes to the *OX Connector* app. It follows the [Keep a Changelog](https://keepachangelog.com/en/1.0.0/)⁴⁸ format and adheres to [Semantic Versioning](https://semver.org/spec/v2.0.0.html)⁴⁹.

9.1.1 v4.0.0

Released: 2026-09-16

Changed

The container image for the OX Connector changed from *Alpine Linux* to a *UCS Base Image*. In most cases, this change has no effect. If manual workflows rely on properties of the underlying OX Connector image, adjust them. The image also updates Python from 3.9 to 3.13.

The *OX Connector* app now uses *Structured logging on UCS* (page 25).

The OX Connector no longer receives changes from the UCS LDAP directory through the App Center *Listener* mechanism. It now subscribes to the Nubus *Provisioning Service* through the *Provisioning API*. The *Provisioning Consumer* inside the app container receives the changes directly, without the intermediate JSON files. The database location changed from `/var/lib/univention-appcenter/apps/ox-connector/data/listener/ox-connector.db` to `/var/lib/univention-appcenter/apps/ox-connector/data/ox-connector.db`. During an update, the app automatically moves existing databases.

The OX Connector depends on the *Provisioning Service*. During installation or upgrade, you can set the provisioning parameters. For more information, see [Provisioning Service](#)⁵⁰.

Removed

The following tools have been removed:

- `rebuild-old.db`
- `remove-from-ox-db-cache`
- `update-ox-db-cache`
- `check_sync_status.py`
- `get_current_error.py`

They used deprecated database calls and didn't work correctly. Use `univention-ox-connector-task-management` instead.

⁴⁸ <https://keepachangelog.com/en/1.0.0/>

⁴⁹ <https://semver.org/spec/v2.0.0.html>

⁵⁰ <https://docs.software-univention.de/manual/5.2/en/domain-ldap/nubus-provisioning-service.html#nubus-provisioning-service>

9.1.2 v3.2.3

Released: 2026-07-06

Added

For *OX Shared Accounts*, the following attributes are now explicitly synchronized:

- Language
- Timezone
- SMTP server
- IMAP server

The *OX Connector* takes these values from its global configuration. You can't change them for individual shared accounts.

9.1.3 v3.2.2

Released: 2026-06-25

Added

The `OX_SHARED_ACCOUNT_IDENTIFIER` (page 20) setting lets you set the name of an *OX Shared Account*.

Fixed

The lookup for existing OX users before their creation could fail in some cases. It used the UDM object's username for the search instead of the configured `OX_USER_IDENTIFIER` (page 19) value.

9.1.4 v3.2.1

Released: 2026-06-10

Added

Added a migration script for *OX Functional Accounts* to *OX Shared Accounts*. Use the script to migrate existing functional accounts to shared accounts. For more information, see *Migration from functional accounts to shared accounts* (page 39).

9.1.5 v3.2.0

Released: 2026-05-22

Added

Added support for *OX Shared Accounts*. You can now create UDM objects for shared accounts and their corresponding permissions. For more information, see *Shared accounts* (page 38).

9.1.6 v3.1.0

Released: 2026-03-17

Removed

The `groups` property in the *Functional Accounts* module has been removed. Adding groups to *Functional Accounts* was never supported, so it wasn't available in the Univention Management Console (UMC). The logic that still made it possible to set groups through direct Univention Directory Manager access has been removed.

Important

If groups were used as *Functional Account* members, add their members directly to the *Functional Account*. Attempting to add a group to a *Functional Account* will raise an error in UDM after this update.

9.1.7 v3.0.2

Released: 2026-03-12

Fixed

Users can grant deputy permissions to other users through `oxDeputyPermissionGivenTo`. The OX Connector now updates these references when the affected user moves in LDAP.

9.1.8 v3.0.1

Released: 2025-10-06

Changed

The sub-command `/usr/sbin/univention-ox-connector-task-management resync-item` can now resynchronize items from the morgue and the old table.

After consecutive errors, the *OX Connector* increases the delay between runs. This prevents log files from filling with repeated error messages. The delay increases with each consecutive error, up to 20 minutes.

Added

Added the sub-command `/usr/sbin/univention-ox-connector-task-management rewrite-ox-db-id`.

Added the sub-command `/usr/sbin/univention-ox-connector-task-management export-old`.

9.1.9 v3.0.0

Released: 2025-09-29

Changed

The *OX Connector App* now stores its task queue in an SQLite database instead of JSON files. The database stores pending tasks and data for objects that the connector has already processed. GDBM-based key-value stores have been removed. Some helper scripts no longer work: `rebuild-old.db`, `remove-from-ox-db-cache`, and `check_sync_status.py`. Existing monitoring plugins might need adjustment.

The app automatically migrates existing data, but the migration can take time depending on your environment.

Added

You can configure the *OX Connector App* to continue processing the queue after an error. By default, it stops after the first error, so its behavior remains unchanged.

For more information, see [OX_CONNECTOR_STOP_ON_ERROR](#) (page 21).

9.1.10 v2.3.5

Released: 12. August 2025

Removed

The resource manager field has been removed from the UMC. The property remains available on OX resources in UDM, where you can view and edit it.

There are no functional changes in the application.

9.1.11 v2.3.4

Released: 23. July 2025

Fixed

Fixed the OX Resources UDM handler, which prevented the OX Connector from working with UCS 5.2-2.

9.1.12 v2.3.3

Released: 10. June 2025

Fixed

UDM now prevents you from creating an OX Context with an ID that another context already uses.

Changed

Creating a new user in OX can now convert an existing OX guest account with the same email address. Note that this requires OX 8.36.36, which isn't currently present in the *App Center*. Also note that this only works for creating users; modifying users (or similar) won't have this feature. If the *OX Connector* can't send this `convertguest` flag, the old behavior applies: Guest accounts with the same email address as the user being processed will block further processing until the error is resolved.

9.1.13 v2.3.2

Released: 4. June 2025

Fixed

The *OX Connector App* uses an internal key-value store to track object states. The keys are the distinguished names (DNs) of LDAP objects. Previously, the connector stored the DN as provided. It now normalizes and converts them to lowercase. The app normalizes existing key-value store keys during an upgrade. Use **`univention-app shell ox-connector rebuild-old.db`** for this migration.

When you changed a user with the same name as a context administrator through UDM or UMC, an edge case could cause the *OX Connector* to change the OX context administrator in the OX database. This could cause authorization and synchronization issues for the OX context administrator.

Changed

In certain cases, changes to an OX user weren't immediately reflected in OX, so the *OX Connector* couldn't retrieve the correct database ID. The *OX Connector App* now retries the lookup several times.

9.1.14 v2.3.1

Released: 16. Apr 2025

Added

You can now specify a log level for the *OX Connector*. For more information, see [App settings](#) (page 17).

Changed

The app setting `OX_CONNECTOR_LOG_LEVEL` (page 20) is used to specify the log level of the ox-connector.

Fixed

The `oxContext` attribute syntax has changed from a string to an integer.

9.1.15 v2.3.0

Released: 27. Mar 2025

Added

Added support for provisioning [OX deputy permissions](#)⁵¹ through the *OX Connector App*. OX deputy permissions let a user act on behalf of another user in OX App Suite, which provides delegated access to email and calendars. Administrators can configure these permissions to control the access level and actions that deputies can perform.

You can now add external CA certificates to the container. For more information, see [Import additional CA certificates on UCS](#) (page 25).

Changed

The app setting `OX_IMAP_LOGIN` (page 18) can contain all attribute names as placeholders. While the *OX Connector* still replaces the default value `{}` with the user's email address, you can set it to `{univentionObjectIdentifier}`, for example, given that such an attribute exists.

Fixed

Translations now work correctly.

9.1.16 v2.2.15

Released: 17. Feb 2025

Changed

Improved the error message when the `OX_USER_IDENTIFIER` (page 19) or `OX_GROUP_IDENTIFIER` (page 20) app setting isn't configured correctly.

Improved the error message when an object requires re-provisioning.

Fixed

The documentation wasn't explicit about setting the administrative password in the app settings for the *OX Connector App*.

The internal key-value store that tracks user objects now handles DN keys case-insensitively.

9.1.17 v2.2.14

Released: 29. Oct 2024

Changed

Access profile names can now include special characters.

⁵¹ https://documentation.open-xchange.com/8/middleware/permissions_and_capabilities/deputy_permission.html

9.1.18 v2.2.13

Released: 17. Sep 2024

Changed

Previously, the OX Connector didn't change the `default_sender_address` user preference when you changed an OX user in UDM. When a UDM change updates `primaryMailAddress`, and the previous address was the user's `default_sender_address`, the OX Connector updates `default_sender_address` to the new email address.

9.1.19 v2.2.12

Released: 28. Aug 2024

Changed

You can now add LDAP containers to the list of default containers for functional accounts and select the container before creating a new functional account in UMC. For more information, see the [Functional accounts](#) (page 37).

9.1.20 v2.2.11

Released: 23. May 2024

Fixed

Fixed a bug that prevented the removal of Open-Xchange contexts.

9.1.21 v2.2.10

Released: 26. April 2024

Changed

The performance of the OX Connector has been improved.

9.1.22 v2.2.9

Released: 12. April 2024

Added

You can now change the attribute mapping between Open-Xchange and UCS through the `change_attribute_mapping.py` script. For more information, see [User attribute mapping](#) (page 22).

9.1.23 v2.2.8

Released: 16. January 2024

Changed

The `meta.db` file also stores the error message and the filename associated with the error.

Added

The `get_current_error.py` script outputs a JSON object containing the contents of the `meta.db` file. You can use this output to automate app health checks.

The app settings `OX_USER_IDENTIFIER` (page 19) and `OX_GROUP_IDENTIFIER` (page 20) have been added. They give control over which UDM property is used as the unique identifier for users and groups in *OX App Suite*.

Added the `check_sync_status.py` script. You can use it to identify data inconsistencies between UDM, OX App Suite, and listener files.

9.1.24 v2.2.7

Released: 7. September 2023

Changed

The `OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE` (page 18) app setting can now contain any string, which simplifies SSO configurations.

Fixed

Fixed handling of an empty `OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE` (page 18) app setting, see [Bug #56523](#)⁵².

Fixed an error when you changed both the context and username in the same operation, see [Bug #56525](#)⁵³.

9.1.25 v2.2.6

Released: 18. August 2023

Changed

The *Functional Account* login field is now configurable through the app setting `OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE` (page 18).

9.1.26 v2.2.5

Released: 16. August 2023

Changed

User context changes now use the `UserCopy` service.

9.1.27 v2.2.4

Released: 13. July 2023

Changed

The `imaplogin` field is now configurable through the app setting `OX_IMAP_LOGIN` (page 18).

9.1.28 v2.2.3

Released: 27. June 2023

Fixed

Corrected a typo in the `listener_trigger` script.

9.1.29 v2.2.2

Released: 22. June 2023

Fixed

The *OX Connector App* now prevents values set by users in the *App Suite* app from being overwritten incorrectly.

⁵² https://forge.univention.org/bugzilla/show_bug.cgi?id=56523

⁵³ https://forge.univention.org/bugzilla/show_bug.cgi?id=56525

9.1.30 v2.2.1

Released: 07. June 2023

Changed

You can no longer change a group's OX context in the UMC groups module. The OX Connector derives the context from the group's users.

9.1.31 v2.2.0

Released: 01. June 2023

Changed

Removed use of old `oxDrive` and `oxAccessUSM` UDM properties. The *OX Connector* only uses the `oxmail/accessprofile` objects to control access rights.

The *OX Connector* now supports duplicate `oxDisplayName` values.

The *OX Connector* only sets a user's `default_sender_address`, `language`, and `timezone` when initially creating a user. Afterwards, any user can configure their settings in the *OX App Suite* front-end.

The OX connector can handle user files in `listener/old/` without the `oxContext` attribute.

Deprecated

`oxTimeZone` and `oxLanguage` remain available as UDM attributes, but the *OX Connector* no longer evaluates them. It uses the values from the app settings instead.

`oxDisplayName` remains available and is evaluated. A later version will use a user's original `displayName` value.

9.1.32 v2.1.4

Released: 31. May 2023

This version has been revoked

9.1.33 v2.1.3

Released: 21. April 2023

Fixed

Changes to the `oxAccessUSM` attribute are now considered by the provisioning logic.

Changed

Added a helper script that removes old listener files for users with an empty `oxContextIDNum` attribute.

Removed `bindpwd` uses from `createextattr.py` script, see [Bug #55985](#)⁵⁴.

9.1.34 v2.1.2

Released: 4. April 2023

Changed

Updated the `inst` script for compatibility with the App Center *OX App Suite*.

⁵⁴ https://forge.univention.org/bugzilla/show_bug.cgi?id=55985

9.1.35 v2.1.1

Released: 9. December 2022

Fixed

Fixed a bug that prevented users from creating OX users in the Univention Management Console.

9.1.36 v2.1.0

Released: 14. November 2022

Fixed

Removed the unnecessary `gid_ox` syntax for OX group names. All valid group names in UCS are now accepted in OX.

Avoided an unnecessary `group change` operation that can fail for large groups and cause the *OX Connector* to repeatedly try to delete an already deleted user.

Changed the `oxcontext contextid` syntax from a string to an integer.

Changed

Refactored the internal project structure.

Updated scripts and internal files.

Added

Prepared support for Univention *OX App Suite*.

9.1.37 v2.0.1

Released: 9. September 2022

Fixed

Avoided unnecessary OX database lookups when synchronizing groups. The OX Connector treats users that aren't in the database as absent without performing a second lookup.

Avoided 500 log messages in OX by checking that users exist before looking them up.

9.1.38 v2.0.0

Released: 26. April 2022

Added

With OX App Suite 7.10.6 Open-Xchange added *Functional Mailboxes* to OX App Suite, see *OX App Suite - Minor Release v7.10.6 - Feature Overview* [12]. OX App Suite shares functional mailboxes among other users in the same context.

For more information, see *Functional accounts* (page 37).

9.1.39 v1.1.0

Added

OX App Suite supports access rights and can grant them individually to users. The *OX Connector App* supports *access profiles* through the `ModuleAccessDefinitions.properties` file.

The connector generates the file locally on the UCS system each time an administrator modifies objects in the Uni-vention Directory Manager module `oxmail/accessprofile`. It doesn't provision the data to *OX App Suite* directly. The *OX Connector* uses the *access profiles* and sets the attribute `oxAccess` during provisioning.

For limitations, see *No validation of access profile rights* (page 54).

9.2 Changelog OX Connector on Kubernetes

Find information about deploying *OX Connector Provisioning Consumer* for user provisioning at *Install on Nubus for Kubernetes* (page 7). This changelog documents all notable changes to *OX Consumer*.

This project follows *Keep a Changelog*⁵⁵ format and *Semantic Versioning*⁵⁶.

9.2.1 v0.41.1

Released: 2026-08-03

Fixed

Relaxed database integrity checks when adding relations between objects. If objects encounter synchronization errors and can't be saved, the relation is still added. The library layer enforces integrity checks instead of the DBMS.

If you deployed v0.41.0 and your database contains these foreign-key constraints, remove them with the following command: **ALTER TABLE relations DROP CONSTRAINT relations_src_obj_id_fkey; ALTER TABLE relations DROP CONSTRAINT relations_dst_obj_id_fkey;** Use equivalent commands for other database systems.

9.2.2 v0.41.0

Released: 2026-07-10

Migration

When you *upgrade from a previous version to v0.41.0* (page 84), update your configuration so that the `init-db` container can complete the following procedure. The connector performs a full synchronization during its first startup, which can take some time.

1. Detect empty database tables.
2. Recreate the subscription to the *Provisioning Service*.
3. Trigger a full synchronization of all UDM objects to the *OX Connector*.
4. Create the cache in the database.

Warning

Before you upgrade from a previous version to `v0.41.0`, complete the migration steps. Otherwise, the *OX Connector* fails during startup.

Go through the following steps to update your configuration:

1. Create a PostgreSQL user and empty database for the *OX Connector*, see *Configure a database* (page 26).
2. Provide a PostgreSQL database connection URL through the `openXchange.oxDbConnectionString` (page 28) Helm value in your custom values file. See *Prepare the OX Consumer configuration* (page 11).

⁵⁵ <https://keepachangelog.com/en/1.0.0/>

⁵⁶ <https://semver.org/spec/v2.0.0.html>

3. Retrieve the administrative credentials for the *Provisioning Service* in your Nubus for Kubernetes deployment. For information about retrieving these credentials, refer to [Obtain administrative credentials⁵⁷](#) in *Univention Nubus - Customization and Modification Manual* [5].

Tip**Use existing Kubernetes secrets**

Instead of specifying passwords directly in the values file, you can reference existing Kubernetes secrets. When you use existing secrets, Helm ignores inline `password` values.

4. Provide the provisioning admin credentials through the following Helm Chart values:
 - `provisioningApi.resync.auth.username` and `provisioningApi.resync.auth.password`
 - or `provisioningApi.resync.auth.existingSecret`
5. Run `helm upgrade` as described in [Listing 3.11](#) in *Install the OX Consumer* (page 14).

Added

This version adds support for *OX Shared Accounts*. You can now add UDM objects for shared accounts and their permissions. For more information, see [Shared accounts](#) (page 38).

OX Shared Accounts require a database. The *OX Connector* now requires a PostgreSQL database. The database isn't included with the packaged *OX App Suite* and *OX Connector* integration. You must provide it separately. For more information, read [Configure a database](#) (page 26).

Warning

Before you upgrade from a previous version to `v0.41.0`, complete the migration steps in [Migration](#) (page 84). Otherwise, this version of the *OX Connector* fails during startup.

Fixed

The `ox-connector` pod now updates and restarts correctly when the `openXchange.auth.password` value changes.

9.2.3 v0.36.2

Released: 2026-03-17

Removed

The `groups` property has been removed from the *Functional Accounts* module. Functional accounts never supported groups, so the Univention Management Console didn't offer this option. The connector no longer lets you set groups through direct Univention Directory Manager access.

Important

If groups were used as *Functional Account* members, add their members directly to the *Functional Account*. Attempting to add a group to a *Functional Account* will raise an error in UDM after this update.

⁵⁷ <https://docs.software-univention.de/nubus-customization/1.x/en/api/provisioning.html#customization-api-provisioning-authentication>

9.2.4 v0.36.1

Released: 2026-03-12

Fixed

The `oxDeputyPermissionGivenTo` property lets users grant deputy permissions to other users. The connector didn't update these references when a referenced user was moved in LDAP. This issue is fixed.

9.2.5 v0.36.0

Released: 4. Mar 2026

Changed

The *OX Consumer* now uses the UDM property `univentionObjectIdentifier` instead of `entryUUID` for the functional account login template. A functional account provides a shared mailbox and calendar for multiple users in *OX App Suite*. If your configuration uses `entryUUID`, consult the *OX App Suite* and **Dovecot** documentation to update the sign-in configuration to use `univentionObjectIdentifier`. Without this update, users can't sign in to newly provisioned functional accounts.

Important

Before you upgrade your production environment to *OX Consumer* 0.36.0, test the update in a test environment and verify that users can sign in to existing and newly created functional accounts.

After you upgrade to *OX Consumer* 0.36.0, validate the following:

- Create a new functional account after the upgrade and verify that it was created successfully.
- Verify that members can access the new functional account.
- If validation fails, don't proceed with upgrading additional environments. Contact your support provider before continuing.

DOCUMENT CHANGELOG

This section shows the history of updates to the *OX Connector for Nubus - Manual*.

10.1 Year 2026

Date	Chapter or topic	Change
25. Sep 2026	<i>Troubleshoot OX Connector on Nubus for Kubernetes</i> (page 68)	Add section about troubleshooting for Nubus for Kubernetes and move the common content about managing provisioning tasks, migration to shared accounts, and collecting support ticket information to separate sections.
24. Sep 2026	<i>Changelog</i> (page 75)	Add the changelog entries for the <i>OX Connector App</i> on Nubus for UCS and for the <i>OX Connector Provisioning Consumer</i> on Nubus for Kubernetes.
21. Sep 2026	<i>Troubleshoot OX Connector on Nubus for UCS</i> (page 63)	Add section about troubleshooting for Nubus for UCS.
17. Sep 2026	<i>Limitations</i> (page 53)	Add description for the limitations of the OX Connector.
9. Sep 2026	<i>For Nubus for Kubernetes</i> (page 48)	Add architecture description for OX Connector on Nubus for Kubernetes.
27. Aug 2026	<i>For Nubus for UCS</i> (page 44)	Add architecture description for OX Connector on Nubus for UCS.
20. Aug 2026	<i>Usage</i> (page 35)	Add instructions for managing OX App Suite contexts, users, groups, access profiles, functional accounts, resources, and shared accounts. Add instructions for migrating functional accounts to shared accounts.
12. Aug 2026	<i>Configuration for Nubus for Kubernetes</i> (page 26)	Add generated Helm Chart values reference for the <i>OX Connector</i> on Nubus for Kubernetes.
10. Aug 2026	<i>Configuration reference</i> (page 17)	Add UCS configuration reference for the <i>OX Connector</i> app.
27. Jul 2026	<i>Install on Nubus for Kubernetes</i> (page 7)	Add installation instructions for the <i>OX Connector</i> on Nubus for Kubernetes.
24. Jul 2026	<i>Installation</i> (page 5)	Add installation instructions for the <i>OX Connector</i> app on Nubus for UCS.
23. Jul 2026	<i>Prerequisites</i> (page 3)	Add prerequisites section for the <i>OX Connector</i> .
22. Jul 2026	<i>OX Connector for Nubus</i> (page 1)	Add introduction section for the document.
22. Jul 2026	Document	Initial document setup

BIBLIOGRAPHY

BIBLIOGRAPHY

- [1] *Nubus for UCS 5.2 - Operation Manual*. Univention GmbH, 2025-2026. URL: <https://docs.software-univention.de/ucs-operation/5.2/en/>.
- [2] *Nubus for Kubernetes - Operation Manual 1.x*. Univention GmbH, 2024-2026. URL: <https://docs.software-univention.de/nubus-kubernetes-operation/1.x/en/>.
- [3] *App Suite Admin Guide 7.10.6*. OX Software GmbH, 2021. URL: https://wiki.open-xchange.com/wiki/index.php?title=AppSuite:AdminGuide_7.10.6.
- [4] *Referential Integrity in OpenLDAP Software 2.4 Administrator's Guide*. The OpenLDAP Project, March 2021. URL: <https://openldap.org/doc/admin24/overlays.html#Referential%20Integrity>.
- [5] *Univention Nubus - Customization and Modification Manual*. Univention GmbH, 2024-2026. URL: <https://docs.software-univention.de/nubus-customization/latest/en/>.
- [6] *Nubus Manual 1.x*. Univention GmbH, 2025-2026. URL: <https://docs.software-univention.de/nubus-manual/1.x/en/>.
- [7] *App Suite Context management*. OX Software GmbH, 2020. URL: https://wiki.open-xchange.com/wiki/index.php?title=AppSuite:Context_management.
- [8] *App Suite Resource management*. OX Software GmbH, 2015. URL: https://wiki.open-xchange.com/wiki/index.php?title=AppSuite:Resource_management.
- [9] *Middleware Documentation*. OX Software GmbH, 2026. URL: <https://documentation.open-xchange.com/8/middleware/>.
- [10] *Univention Corporate Server Architecture*. Univention GmbH, 2023-2024. URL: <https://docs.software-univention.de/architecture/latest/en/>.
- [11] *Nubus for Kubernetes - Architecture Manual 1.x*. Univention GmbH, 2024-2026. URL: <https://docs.software-univention.de/nubus-kubernetes-architecture/1.x/en/>.
- [12] *OX App Suite - Minor Release v7.10.6 - Feature Overview*. OX Software GmbH, December 2021. URL: https://www.open-xchange.com/hubfs/Imported%20files/Feature_Overview_OXAppSuite_7_10_6.pdf?hsLang=en.

A

access profiles
 changelog, 83

B

Bugzilla
 Bug #55985, 82
 Bug #56523, 81
 Bug #56525, 81

C

change_attribute_mapping.py command line
 option
 dump, 22
 modify, 22, 23
 restore_default, 22
changelog
 access profiles, 83
 functional mailbox, 83

D

DEFAULT_CONTEXT, 21, 22
Directory Manager, 43
dump
 change_attribute_mapping.py command
 line option, 22

E

environment variable
 affinity, 27
 DEFAULT_CONTEXT, 17, 21, 22
 environment, 27
 extraVolumeMounts, 27
 extraVolumes, 27
 fullnameOverride, 27
 global.imagePullPolicy, 27
 global.imagePullSecrets, 27
 global.imageRegistry, 27
 LOCAL_TIMEZONE, 18
 nameOverride, 28
 nodeSelector, 28
 openXchange.auth.existingSecret.keyMapping.password, 28
 openXchange.auth.existingSecret.name, 28
 openXchange.auth.password, 13, 28

openXchange.auth.username, 28
openXchange.domainName, 13, 28
openXchange.logLevel, 28
openXchange.mappings.groupIdentifier, 20, 28
openXchange.mappings.sharedAccountIdentifier, 20, 28
openXchange.mappings.userIdentifier, 19, 28
openXchange.oxDbConnectionString, 13, 26, 28, 84
openXchange.oxDefaultContext, 28
openXchange.oxDeputyPermissions, 29
openXchange.oxImapServer, 13, 29
openXchange.oxLanguage, 29
openXchange.oxLocalTimezone, 29
openXchange.oxSharedAccount, 29
openXchange.oxSmtpServer, 13, 29
openXchange.oxSoapServer, 13, 29
ox/context/id, 21, 22
OX_CONNECTOR_LOG_LEVEL, 20, 26, 79
OX_CONNECTOR_STOP_ON_ERROR, 21, 53, 54, 77
OX_ENABLE_DEPUTY_PERMISSIONS, 20
OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE, 18, 81
OX_GROUP_IDENTIFIER, 20, 66, 71, 79, 80
OX_IMAP_LOGIN, 18, 79, 81
OX_IMAP_SERVER, 13, 17
OX_LANGUAGE, 18
OX_MASTER_ADMIN, 3, 18
OX_MASTER_PASSWORD, 3, 13, 18
OX_SHARED_ACCOUNT_IDENTIFIER, 20, 76
OX_SMTP_SERVER, 13, 17
OX_SOAP_SERVER, 13, 17
OX_USER_IDENTIFIER, 19, 66, 71, 76, 79, 80
oxConnector.extraEnvVars, 29
oxConnector.image.pullPolicy, 29
oxConnector.image.registry, 29
oxConnector.image.repository, 29
oxConnector.image.tag, 29
persistence.size, 30
persistence.storageClass, 30
podAnnotations, 30
podSecurityContext.fsGroup, 30
podSecurityContext.runAsGroup, 30
podSecurityContext.runAsNonRoot, 30

`podSecurityContext.runAsUser`, 30
`podSecurityContext.seccompProfile.type`, 30
`probes.liveness.exec.command`, 30
`probes.liveness.failureThreshold`, 30
`probes.liveness.initialDelaySeconds`, 30
`probes.liveness.periodSeconds`, 30
`probes.liveness.successThreshold`, 30
`probes.liveness.timeoutSeconds`, 30
`probes.readiness.exec.command`, 30
`probes.readiness.failureThreshold`, 30
`probes.readiness.initialDelaySeconds`, 30
`probes.readiness.periodSeconds`, 30
`probes.readiness.successThreshold`, 31
`probes.readiness.timeoutSeconds`, 31
`provisioningApi.auth.existingSecret.keyMapping.password`, 31
`provisioningApi.auth.existingSecret.name`, 31
`provisioningApi.auth.password`, 13, 31
`provisioningApi.auth.username`, 13, 31
`provisioningApi.config.maxAcknowledgementRetries`, 31
`provisioningApi.connection.baseUrl`, 13, 31
`provisioningApi.resync.auth.existingSecret.keyMapping.password`, 31
`provisioningApi.resync.auth.existingSecret.name`, 31
`provisioningApi.resync.auth.password`, 13, 31
`provisioningApi.resync.auth.username`, 31
`provisioningApi.resync.enabled`, 13, 31
`replicaCount`, 32
`resources.limits.cpu`, 32
`resources.limits.memory`, 32
`resources.requests.cpu`, 32
`resources.requests.memory`, 32
`resourcesWaitForDependency.limits.cpu`, 32
`resourcesWaitForDependency.limits.memory`, 32
`resourcesWaitForDependency.requests.cpu`, 32
`resourcesWaitForDependency.requests.memory`, 32
`securityContext.allowPrivilegeEscalation`, 32
`securityContext.capabilities.drop`, 32
`securityContext.privileged`, 32
`securityContext.readOnlyRootFilesystem`, 32
`securityContext.runAsGroup`, 32
`securityContext.runAsNonRoot`, 32

`securityContext.runAsUser`, 32
`securityContext.seccompProfile.type`, 32
`serviceAccount.annotations`, 33
`serviceAccount.automountServiceAccountToken`, 33
`serviceAccount.create`, 33
`serviceAccount.labels`, 33
`serviceAccount.name`, 33
`tolerations`, 33
`waitForDependency.image.pullPolicy`, 33
`waitForDependency.image.registry`, 33
`waitForDependency.image.repository`, 33
`waitForDependency.image.tag`, 33

F

functional mailbox
changelog, 83

I

Identity Store and Directory Service, 43

M

modify
`change_attribute_mapping.py` command
line option, 22, 23

Morgue, 56

O

Old, 56
`openXchange.auth.password`, 13
`openXchange.domainName`, 13
`openXchange.mappings.groupIdentifier`, 20
`openXchange.mappings.sharedAccountIdentifier`, 20
`openXchange.mappings.userIdentifier`, 19
`openXchange.oxDbConnectionString`, 13, 26, 84
`openXchange.oxImapServer`, 13
`openXchange.oxSmtptServer`, 13
`openXchange.oxSoapServer`, 13
OX App Suite, 44
OX Connector, 43
OX Connector Provisioning Consumer, 44
`ox/context/id`, 21, 22
`OX_CONNECTOR_LOG_LEVEL`, 26, 79
`OX_CONNECTOR_STOP_ON_ERROR`, 53, 54, 77
`OX_ENABLE_DEPUTY_PERMISSIONS`, 20
`OX_FUNCTIONAL_ACCOUNT_LOGIN_TEMPLATE`, 81
`OX_GROUP_IDENTIFIER`, 66, 71, 79, 80
`OX_IMAP_LOGIN`, 79, 81
`OX_IMAP_SERVER`, 13
`OX_MASTER_ADMIN`, 3
`OX_MASTER_PASSWORD`, 3, 13
`OX_SHARED_ACCOUNT_IDENTIFIER`, 76
`OX_SMTP_SERVER`, 13
`OX_SOAP_SERVER`, 13, 17
`OX_USER_IDENTIFIER`, 66, 71, 76, 79, 80

P

PostgreSQL database: OX Connector, [48](#)
Provisioning API, [43](#)
Provisioning Service, [43](#)
`provisioningApi.auth.password`, [13](#)
`provisioningApi.auth.username`, [13](#)
`provisioningApi.connection.baseUrl`, [13](#)
`provisioningApi.resync.auth.password`, [13](#)
`provisioningApi.resync.enabled`, [13](#)

R

`restore_default`
 `change_attribute_mapping.py` command
 line option, [22](#)

S

SOAP API, [44](#)
Subscription, [48](#)

T

Tasks, [56](#)

U

`udm modules`
 `oxmail/accessprofile`, [83](#)
 `oxmail/functional_account`, [83](#)