univention
be open.

Univention Corporate Server - Manual for users and administrators

Release 5.0

Nov 28, 2024

The source of this document is licensed under GNU Affero General Public License v3.0 only.

CONTENTS

1 Introduction **1**
- 1.1 What is Univention Corporate Server? . . . 1
- 1.2 What is Univention Nubus? . . . 2
- 1.3 Overview of UCS . . . 2
- 1.4 Further documentation . . . 6
- 1.5 Symbols and conventions used in this manual . . . 6

2 Installation **9**
- 2.1 Selecting the installation mode . . . 9
- 2.2 Selecting the installation language . . . 11
- 2.3 Selecting the location . . . 12
- 2.4 Selecting the keyboard layout . . . 12
- 2.5 Network configuration . . . 12
- 2.6 Setting up the root password . . . 17
- 2.7 Partitioning the hard drive . . . 17
- 2.8 Domain settings . . . 19
- 2.9 Confirming the settings . . . 24
- 2.10 Troubleshooting for installation problems . . . 27
- 2.11 Installation in text mode . . . 27
- 2.12 Installation in the Amazon EC2 cloud . . . 27
- 2.13 Installation in VMware . . . 27

3 Domain services / LDAP directory **29**
- 3.1 Joining domains . . . 29
- 3.2 UCS system roles . . . 33
- 3.3 LDAP directory . . . 36
- 3.4 Listener/notifier domain replication . . . 40
- 3.5 SSL certificate management . . . 43
- 3.6 Kerberos . . . 44
- 3.7 Password hashes in the directory service . . . 44
- 3.8 SAML identity provider . . . 45
- 3.9 OpenID Connect Provider . . . 48
- 3.10 Converting a Backup Directory Node backup to the new Primary Directory Node . . . 49
- 3.11 Fault-tolerant domain setup . . . 51
- 3.12 Protocol of activities in the domain . . . 51

4 UCS web interface **55**
- 4.1 Introduction . . . 56
- 4.2 Login . . . 57
- 4.3 UCS portal page . . . 65
- 4.4 Consent for using Cookies . . . 68
- 4.5 Univention Management Console modules . . . 69
- 4.6 LDAP directory browser . . . 73
- 4.7 Policies . . . 74

4.8 Expansion of UMC modules with extended attributes . . . 76
4.9 Structuring of the domain with user-defined LDAP structures . . . 80
4.10 Delegated administration for UMC modules . . . 81
4.11 Command line interface of domain management (Univention Directory Manager) . . . 81
4.12 HTTP API of domain management . . . 88
4.13 Evaluation of data from the LDAP directory with Univention Directory Reports . . . 88
4.14 Let's Encrypt . . . 89

5 Software deployment **91**
5.1 Differentiation of update variants / UCS versions . . . 91
5.2 Univention App Center . . . 92
5.3 Updates of UCS systems . . . 96
5.4 Configuration of the repository server for updates and package installations . . . 98
5.5 Installation of further software . . . 99
5.6 Specification of an update point using the package maintenance policy . . . 102
5.7 Central monitoring of software installation statuses with the software monitor . . . 103

6 User management **105**
6.1 User management through Univention Management Console module . . . 106
6.2 User activation for apps . . . 114
6.3 User password management . . . 115
6.4 Password settings for Windows clients when using Samba . . . 118
6.5 User self services . . . 119
6.6 Automatic lockout of users after failed login attempts . . . 127
6.7 User templates . . . 129
6.8 Overlay module for recording an account's last successful LDAP bind . . . 130
6.9 Prevent reuse of user property values . . . 131

7 Group management **135**
7.1 User group assignments . . . 135
7.2 Recommendation for group name definition . . . 135
7.3 Managing groups via Univention Management Console module . . . 136
7.4 Group nesting with groups in groups . . . 139
7.5 Local group cache . . . 139
7.6 Synchronization of Active Directory groups when using Samba/AD . . . 140
7.7 Overlay module for displaying the group information on user objects . . . 141

8 Computer management **143**
8.1 Management of computer accounts via Univention Management Console module . . . 143
8.2 Configuration of hardware and drivers . . . 149
8.3 Administration of local system configuration with Univention Configuration Registry . . . 156
8.4 Basic system services . . . 162

9 Services for Windows **171**
9.1 Operation of a Samba domain based on Active Directory . . . 171
9.2 Active Directory Connection . . . 180
9.3 Migrating an Active Directory domain to UCS using Univention AD Takeover . . . 196
9.4 Trust relationships . . . 200

10 Identity Management connection to cloud services **203**
10.1 Microsoft 365 Connector . . . 203
10.2 Google Apps for Work Connector . . . 207

11 IP and network management **211**
11.1 Network objects . . . 211
11.2 Administration of DNS data with BIND . . . 212
11.3 IP assignment via DHCP . . . 220
11.4 Packet filter with Univention Firewall . . . 226
11.5 Web proxy for caching and policy management / virus scan . . . 227

11.6 RADIUS . . . 229

12 File share management **237**
12.1 Access rights to data in shares . . . 237
12.2 Management of shares via UMC module . . . 238
12.3 Support for MSDFS . . . 245
12.4 Configuration of file system quota . . . 245

13 Print services **249**
13.1 Installing a print server . . . 249
13.2 Setting the local configuration properties of a print server . . . 250
13.3 Creating a printer share . . . 250
13.4 Creating a printer group . . . 253
13.5 Administration of print jobs and print queues . . . 254
13.6 Generating PDF documents from print jobs . . . 255
13.7 Mounting of print shares in Windows clients . . . 255
13.8 Integrating additional PPD files . . . 259

14 Mail services **261**
14.1 Installation . . . 262
14.2 Management of the mail server data . . . 262
14.3 Spam detection and filtering . . . 267
14.4 Identification of viruses and malware . . . 267
14.5 Identification of Spam sources with DNS-based Blackhole Lists . . . 268
14.6 Integration of Fetchmail for retrieving mail from external mailboxes . . . 268
14.7 Configuration of the mail server . . . 269
14.8 Configuration of mail clients for the mail server . . . 276
14.9 OX Connector . . . 276

15 Infrastructure monitoring **277**
15.1 UCS Dashboard . . . 277
15.2 Monitoring . . . 280
15.3 Nagios . . . 287

16 Appendix **291**
16.1 Univention Configuration Registry Variables . . . 291
16.2 Bibliography . . . 307
16.3 Indices . . . 307

Bibliography **309**

Index **311**

CHAPTER

ONE

INTRODUCTION

Welcome to the Manual for users and administrators of Univention Corporate Server (UCS). This document is for system administrators who operate UCS.

1.1 What is Univention Corporate Server?

Univention Corporate Server (UCS) is a Linux-based server operating system for the operation and administration of IT infrastructures for companies and authorities. UCS implements an integrated, holistic concept with consistent, central administration. It can ensure the operation of all the components in an interrelated security and trust context, the so-called UCS domain. At the same time, UCS supports a wide range of open standards and includes extensive interfaces to infrastructure components and management tools from other manufacturers, meaning it can integrate with existing environments.

UCS consists of reliable open source software, tried and tested in organizations of different sizes. UCS combines these software components through a unified web interface. This allows the integration and administration of the system in both simple and complex distributed or virtualized environments.

The central functions of UCS are:

- Flexible and extensive identity/infrastructure management for the central administration of servers, workstations, users, and their permissions, server applications and web services.
- Services for integrating the management of existing Microsoft Active Directory domains or even the provision of such services as an alternative for Microsoft-based server systems.
- App Center for simple installation and management of extensions and applications.
- Network and intranet services for administration of DHCP and DNS.
- File and print services.
- Computer administration and monitoring.
- Mail services.

Different software packages in UCS provide these functions covered in detail throughout this manual. Basically, the software packages contained in UCS belong to one of the following main categories:

1. Base system
2. UCS management system with Univention Management Console modules
3. Univention App Center, allowing the installation of further components and applications of other software vendors

The *base system* encompasses the operating system of the UCS Linux distribution maintained by Univention, and based on Debian GNU/Linux. It largely includes the same software selection as Debian GNU/Linux, as well as, additional tools for the installation, updating, and configuration of clients and servers.

The UCS Management System realizes a single point of administration where a single directory service manages the accounts of all domain members, such as users, groups, and hosts, and services, such as DNS, and DHCP. Core components of the management system are the following services:

- OpenLDAP for the directory service
- Samba for provision of domain, file and print services for Windows
- Kerberos for authentication and single sign-on
- DNS for network name resolution
- TLS for secure data transmission between systems

You can use UCS through a web interface, the Univention Management Console modules, or in the command line and in individual scripts. You can extend the UCS management system with APIs (application programming interfaces). UCS provides a flexible client-server architecture which transfers changes to the involved systems and activates them there.

You can install additional components from Univention and other manufacturers through the App Center. They expand the system with numerous functions such as groupware, document management, and services for Windows, meaning that you can also operate them on a UCS system and administer them through the UCS management system.

1.2 What is Univention Nubus?

Univention Nubus is an open source solution for integrating identity and access management of various applications. It offers the following functions:

- administration of users and groups
- a portal with integrated user self service as the first entry port for end users
- numerous interfaces for the integration of applications
- shared single sign-on

Standard integrations based on these interfaces connect common applications. Most components of Nubus originate directly from UCS.

Nubus is the Univention product for identity and access management and the portal. You can deploy Nubus as part of UCS or on a Kubernetes cluster. Univention Corporate Server (UCS) is a way to deploy Nubus with services and integrations on hardware or virtual machines.

1.3 Overview of UCS

Linux is an operating system which always had a focus on stability, security and compatibility with other operating systems. Therefore Linux is predestined for being used in server operating systems that are stable, secure and highly available.

Built on that base, UCS is a server operating system which is optimized for the simple and secure operation and management of applications and infrastructure services in enterprises and public authorities. For efficient and secure management such applications rely on the tight integration in the user and permission management of the UCS Management System.

UCS can be employed as the basis for the IT infrastructure in companies and authorities and provide the central control for it. This makes a considerable contribution to secure, efficient and cost-effective IT operation. The business-critical applications are integrated in a uniform concept, adapted to each other and pre-configured for professional utilization. Alternatively it can be operated as part of an existing Microsoft Active Directory domain.

1.3.1 Commissioning

The use of UCS begins either with a classic operating system installation on a physical server or as a virtual machine. Further information can be found in *Installation* (page 9).

1.3.2 Domain concept

In an IT infrastructure managed with UCS, all servers, clients and users are contained in a common security and trust context, referred to as the UCS domain. Every UCS system is assigned a so-called server role during the installation. Possible system roles are Directory Node, Managed Node and client.

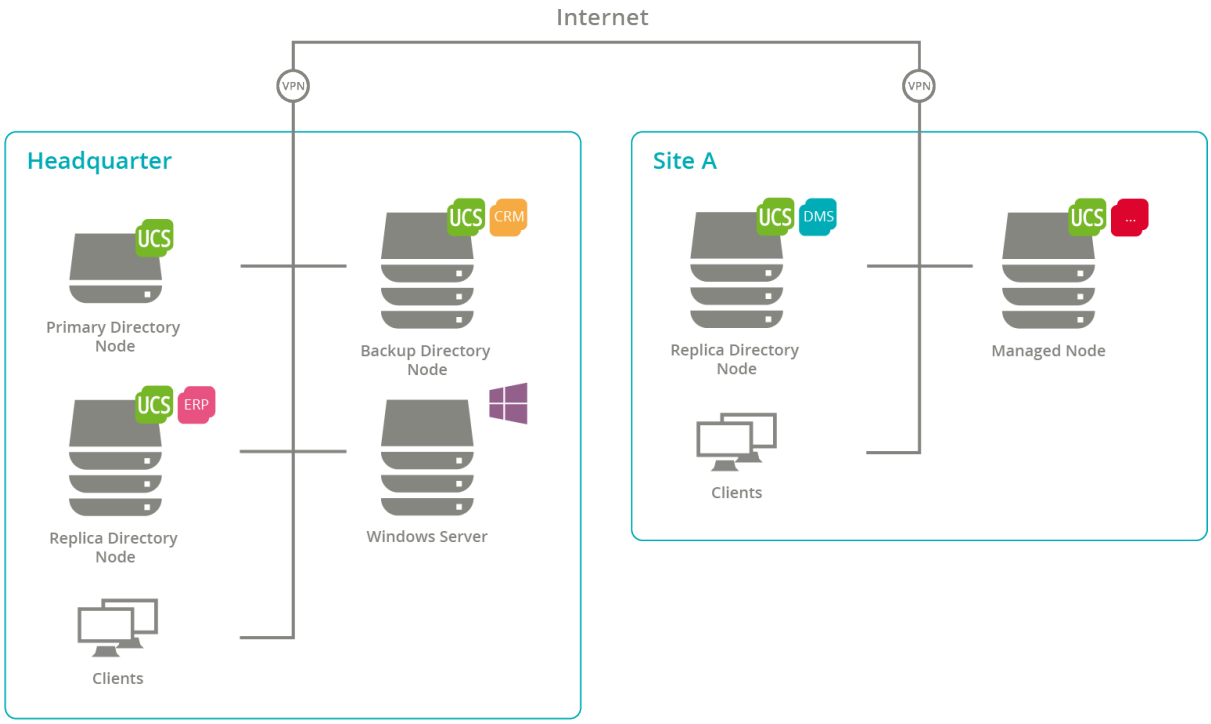


Fig. 1.1: UCS domain concept

Depending on the system role within the domain, such services as Kerberos, OpenLDAP, Samba, modules for domain replication or a Root CA (certification authority) are installed on the computer. These are automatically configured for the selected system role. The manual implementation and configuration of every single service and application is therefore not required. Due to the modular design and extensive configuration interfaces, tailor-made solutions to individual requirements can nevertheless be realized.

The integration of Samba, which provides the domain service for clients and servers operated with Microsoft Windows, makes Univention Corporate Server compatible with Microsoft Active Directory (AD), whereby the system acts as an Active Directory server for Windows-based systems. Consequently, for example, group policies for Microsoft Windows systems can be administrated in the usual way.

UCS can also be operated as part of an existing Microsoft Active Directory domain. This way, users and groups of the Active Directory domain can access applications from the Univention App Center.

Ubuntu or macOS clients can be integrated in a UCS environment, as well (see *Integration of Ubuntu clients* (page 148)).

1.3.3 Expandability with the Univention App Center

The Univention App Center offers additional UCS components and extensions and a broad selection of business IT software, e.g., groupware and collaboration, file exchange, CRM or backup. These applications can be installed in existing environments with a few clicks and are usually ready to use. In most cases they are directly integrated into the UCS Management System such that they are available as Univention Management Console modules. This provides a central management of data on the domain level and obsoletes the separate management of, e.g., user data in multiple places.

1.3.4 LDAP directory service

With the UCS Management System, all the components of the UCS domain can be centrally administrated across computer, operating system and site boundaries. It thus provides a single point of administration for the domain. One primary element of the UCS management system is an LDAP directory in which the data required across the domain for the administration are stored. In addition to the user accounts and similar elements, the data basis of services such as DHCP is also saved there. The central data management in the LDAP directory avoids not only the repeated entry of the same data, but also reduces the probability of errors and inconsistencies.

An LDAP directory has a tree-like structure, the root of which forms the so-called basis of the UCS domain. The UCS domain forms the common security and trust context for its members. An account in the LDAP directory establishes the membership in the UCS domain for users. Computers receive a computer account when they join the domain. Microsoft Windows systems can also join the domain such that users can sign in there with their domain passport.

UCS utilizes OpenLDAP as a directory service server. The directory is provided by the Primary Directory Node and replicated on all UCS Directory Nodes in the domain. The complete LDAP directory is also replicated on a Backup Directory Node as this can replace the Primary Directory Node in an emergency. In contrast, the replication on Replica Directory Node can be restricted to certain areas of the LDAP directory using ACLs (access control lists) in order to realize a selective replication. For example, this may be desirable if data should only be stored on as few servers as possible for security reasons. For secure communication of all systems within the domain, UCS integrates a root CA (certification authority).

Further information can be found in *LDAP directory* (page 36).

1.3.5 Domain administration

Access to the LDAP directory is performed via a web-based user interface through Univention Management Console (UMC) modules. In addition to this, Univention Directory Manager allows the realization of all domain-wide administrative tasks via a command line interface. This is particularly suitable for the integration in scripts or automated administrative steps.

Univention Management Console modules allows to display, edit, delete, and search the data in the LDAP directory via various filter criteria. The web interface offers a range of wizards for the administration of user, groups, networks, computers, directory shares and printers. The administration of computers also comprises comprehensive functions for distributing and updating software. The integrated LDAP directory browser can be used to make further settings and add customer-specific object classes or attributes.

Further information can be found in *UCS web interface* (page 55).

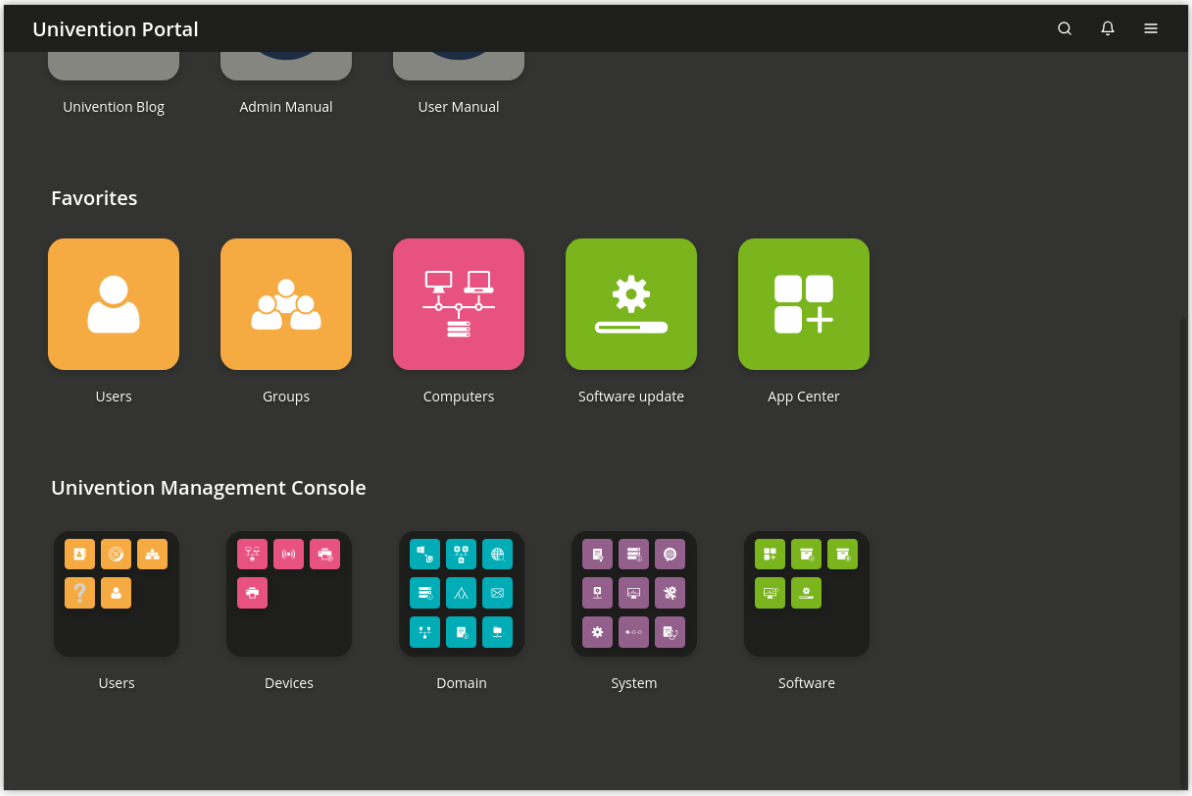


Fig. 1.2: Univention Management Console modules

1.3.6 Computer administration

Univention Management Console modules allows not only the access to the LDAP directory, but also the web-based configuration and administration of individual computers. These include the adaptation of configuration data, the installation of software as well as the monitoring and control of services and the operating system itself. With the UCS Management System, domain administration as well as computer and server configuration is possible from any place via a comfortable graphic web interface.

1.3.7 Policy concept

The tree-like structure of LDAP directories is similar to that of a file system It ensures that objects (such as users, computers, etc.) are in one container which itself can be adopted by other containers. The root container is also called the LDAP base object.

Policies describe certain administrative settings which are applied to more than one object. Linked to containers, they facilitate the administration as they are effective for all objects in the container in question as well as the objects in subfolders.

For example, users can be organized in different containers or organizational units (which are a form of containers) depending on which department they belong to. Settings such as the desktop background or accessible programs can then be connected to these organizational units using policies. Subsequently, they apply for all users within the organizational unit in question.

Further information can be found in *Policies* (page 74).

1.3.8 Listener/notifier replication

The listener/notifier mechanism is an important technical component of the UCS Management System. With this, the creation, editing or deleting of entries in the LDAP directory triggers defined actions on the computers in question. For example, the creation of a directory share with the UMC module *Shares* leads to the share firstly being entered in the LDAP directory. The listener/notifier mechanism then ensures that the NFS and Samba configuration files are also expanded accordingly on the selected server and that the directory is created in the file system of the selected server if it does not already exist.

The listener/notifier mechanism can be expanded with modules for further - also customer-specific - procedures. Consequently, it is used by numerous technology partners for the integration of their products in the LDAP directory service and the UCS Management System for example.

Further information can be found in *Listener/notifier domain replication* (page 40).

1.4 Further documentation

This manual addresses just a small selection of the possibilities in UCS. Among other things, UCS and solutions based on UCS provide:

- Comprehensive support for complex server environments and replication scenarios
- Advanced capabilities for Windows environments
- Central network management with DNS and DHCP
- Monitoring systems and networks
- Print server functionalities
- Proxy server

Further documentation related to UCS and further issues is published under *UCS documentation overview* [1].

1.5 Symbols and conventions used in this manual

The manual uses the following symbols:

Caution: Warnings are highlighted.

Note: Notes are also highlighted.

This table describes the functionality of a UMC module:

Table 1.1: Tab DHCP service

Attribute	Description
Name	The unique name of a DHCP service.
Description	An arbitrary description of the service.

Menu entries, button labels, and similar details with actions are printed in *this font face*.

Names are highlighted.

`Computer names`, `LDAP DNs`, **`program names`**, `file names`, `file paths`, internet addresses[2] and `options` are also optically accented.

`Commands and other keyboard input` is accented optically.

```
In addition, excerpts from configuration files, screen output, etc. are
printed as code block.
```

A backslash (\) at the end of a line signifies that the subsequent line feed is not to be understood as an *end of line*. This circumstance may occur, for example, where commands cannot be represented in one line in the manual, yet have to be entered in the command line in one piece without the backslash or with the backslash and a subsequent `Enter`.

The path to a function is represented in a similar way to a file path. *Users* ‣ *Add* means for example, you have to click *Users* in the main menu and *Add* in the submenu.

[2] https://example.com

CHAPTER

TWO

INSTALLATION

The following documentation describes how to install Univention Corporate Server (UCS). The UCS system is installed from the DVD. The installation is interactive and prompts all the necessary system settings in a graphic interface.

The installation DVD is available for the computer architecture *amd64* (64-bit). In addition to support for the widely distributed BIOS systems, the DVD also includes support for the Unified Extensible Firmware Interface (UEFI) standard. The UEFI support on the DVD is also capable of starting systems with activated Secure Boot and installing UCS there.

Note: Please note that simultaneous operation of UCS and Debian on a UEFI system starting with UCS 5.0-0 is not supported. The reason for this is the GRUB boot loader of Univention Corporate Server, which partly uses the same configuration files as Debian. An already installed Debian leads to the fact that UCS cannot be booted (any more) after the installation of or an update to UCS 5.0. A subsequent installation of Debian will also result in UCS 5.0 not being able to boot.

Following installation on hardware or in a virtualization solution, UCS can also be installed on the Amazon EC2 cloud using an AMI image. Further information can be found in *Installation in the Amazon EC2 cloud* (page 27).

The installer's input masks can be operated with the mouse or via the keyboard.

- The `Tab` key can be used to proceed to the next field.
- The key combination of `Shift+Tab` can be used to return to the previous field.
- The `Enter` key is used to assign values to the input field and confirm buttons.
- Within a list or table, the *arrow keys* can be used for navigating between entries.

Note: The *Cancel* button can be used to cancel the current configuration step. An earlier configuration step can then be selected again in the menu that is subsequently shown. Under certain circumstances, subsequent configuration steps cannot be directly selected if the earlier steps have not been completed.

2.1 Selecting the installation mode

After booting the system from the installation medium, the following boot prompt is displayed:

Now you can choose between several installation procedures.

- *Start with default settings* starts the interactive, graphic installation. During the installation, the system requests a number of parameters such as the network settings, hard drive partitions and domain settings for the UCS system to be installed and then performs the installation and the configuration.
- *Start with manual network settings* performs a standard installation, where the network is not configured automatically through DHCP. This is practical on systems, where the network must be setup manually.
- The *Advanced options* submenu offers advanced options for the installation process for selection:

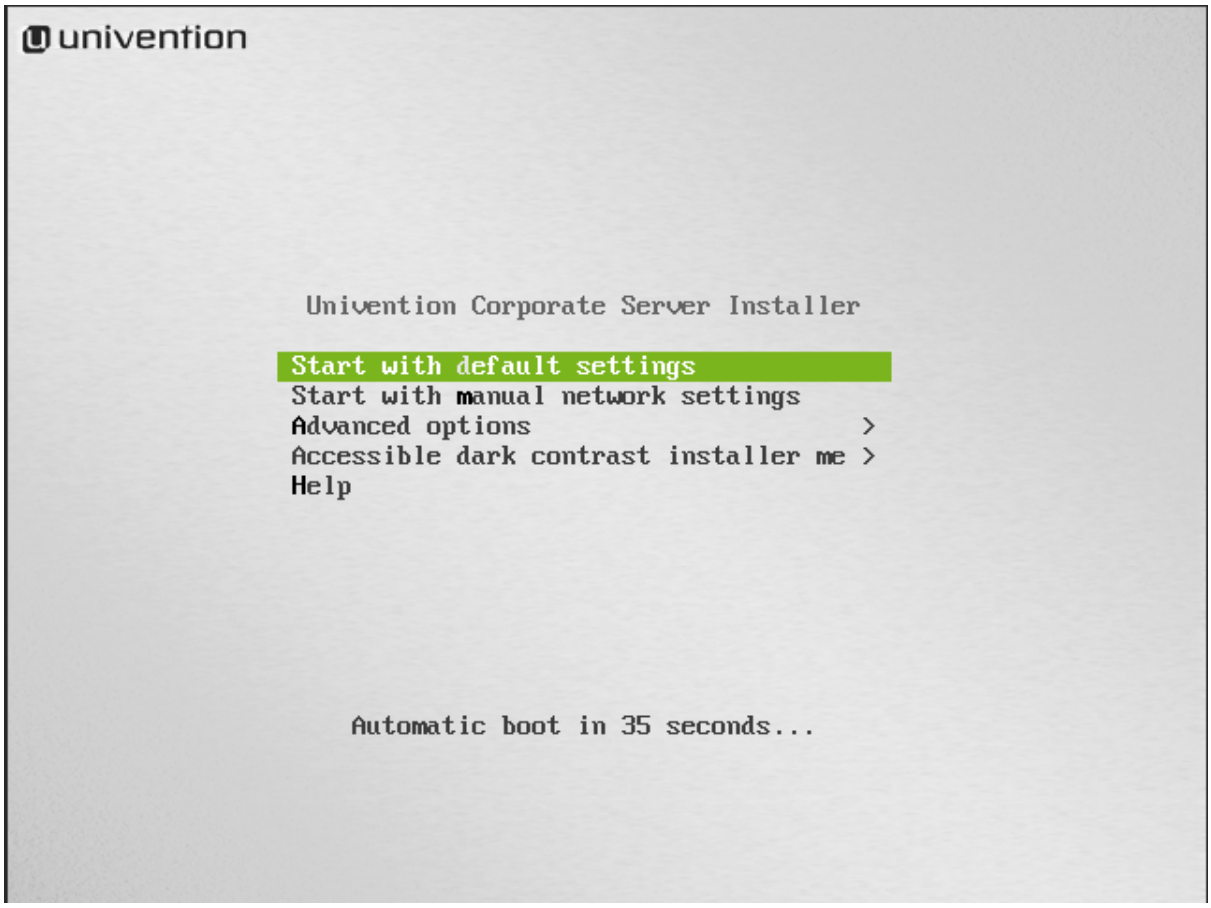


Fig. 2.1: Installation boot prompt

  - *Start in text mode* performs an interactive standard installation in text mode. This is practical on systems which display problems with the graphic version of the installer.
  - *Rescue mode* is there to recover systems unable to boot.
  - *Boot from first hard drive* boots the operating system installed on the first hard drive instead of the UCS installation.
- *Accessible dark contrast installer menu* allows to start the setup in a dark and contrast rich mode.

Once one of the installation option is selected, the kernel is loaded from the installation medium. The actual installation is divided into separate modules, which can be loaded from the installation medium subsequently if necessary. There are modules for network configuration or for selecting the software to be installed, among others.

2.2 Selecting the installation language

In the first step, you can select the system language you wish to use. The selection has an influence on the use of language-specific characters and permits the representation of program output in the selected languages in the installed UCS system.

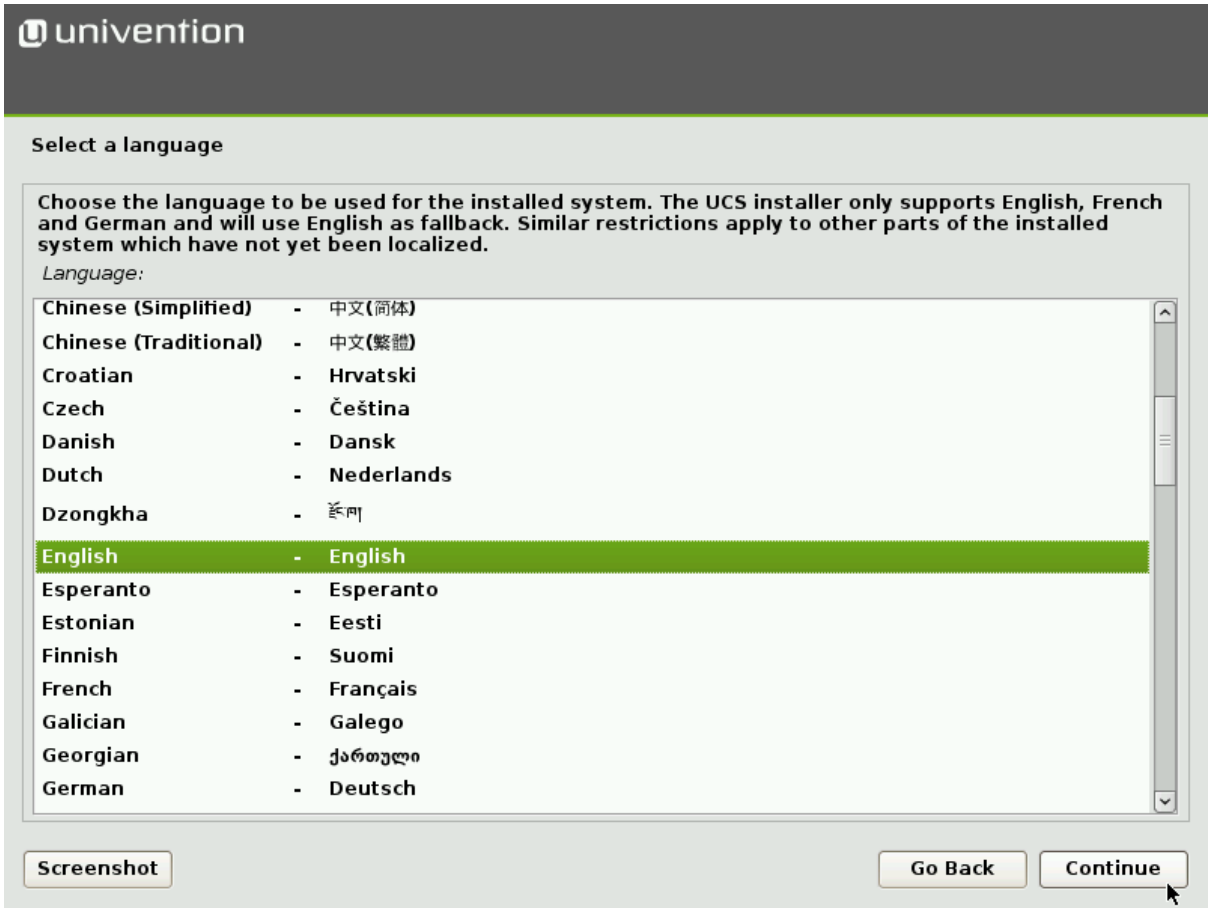


Fig. 2.2: Selecting the installation language

If Univention Installer has been translated into the selected language (currently German and English), the selected language is also used during the installation, otherwise the installation is performed in English.

2.3 Selecting the location

Once the system language has been selected, a small list of locations is displayed based on the selected language. Select a suitable location from the list. The selected location is used to set the time zone or the correct language variant, for example. Should none of the displayed locations be appropriate, a more extensive list can be displayed using the menu entry **other**.

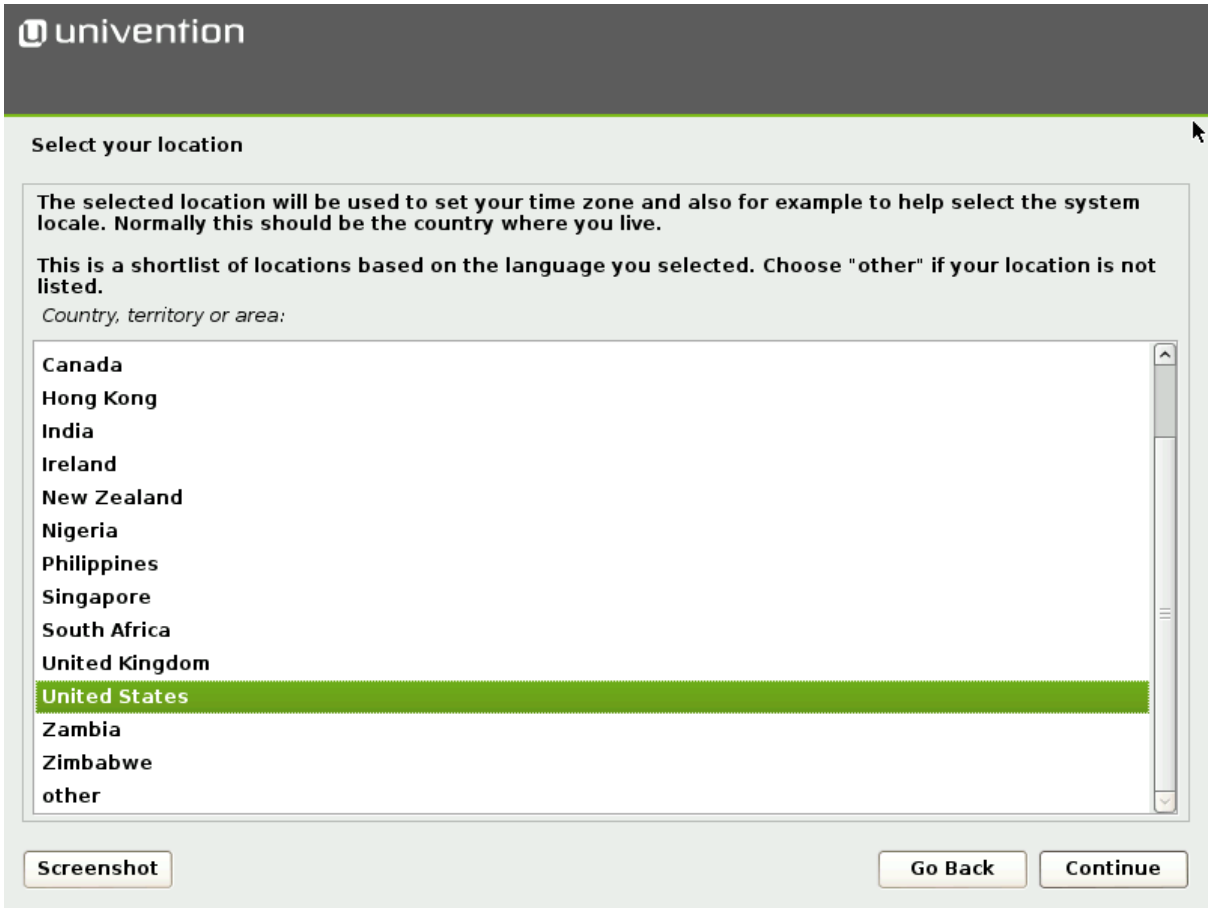


Fig. 2.3: Selecting the location

2.4 Selecting the keyboard layout

The keyboard layout can be selected independently of the system language. The language selected here should be compatible with the keyboard used as it may otherwise cause operating problems.

2.5 Network configuration

Initially, the Univention Installer attempts to configure the network interfaces automatically. This can be disabled by selecting the menu item *Start with manual network settings* from the menu of the boot loader. Firstly, an attempt is made to determine an IPv6 address via the stateless address autoconfiguration (SLAAC). If this is not successful, the Univention Installer attempts to request an IPv4 address via the Dynamic Host Configuration Protocol (DHCP). If this is successful, the manual network configuration of Univention Installer is skipped.

If there is no DHCP server present in the local network or static configuration of the network interface is required, the *Cancel* button can be selected. The Univention Installer then offers to repeat the automatic configuration or to configure the interface manually.

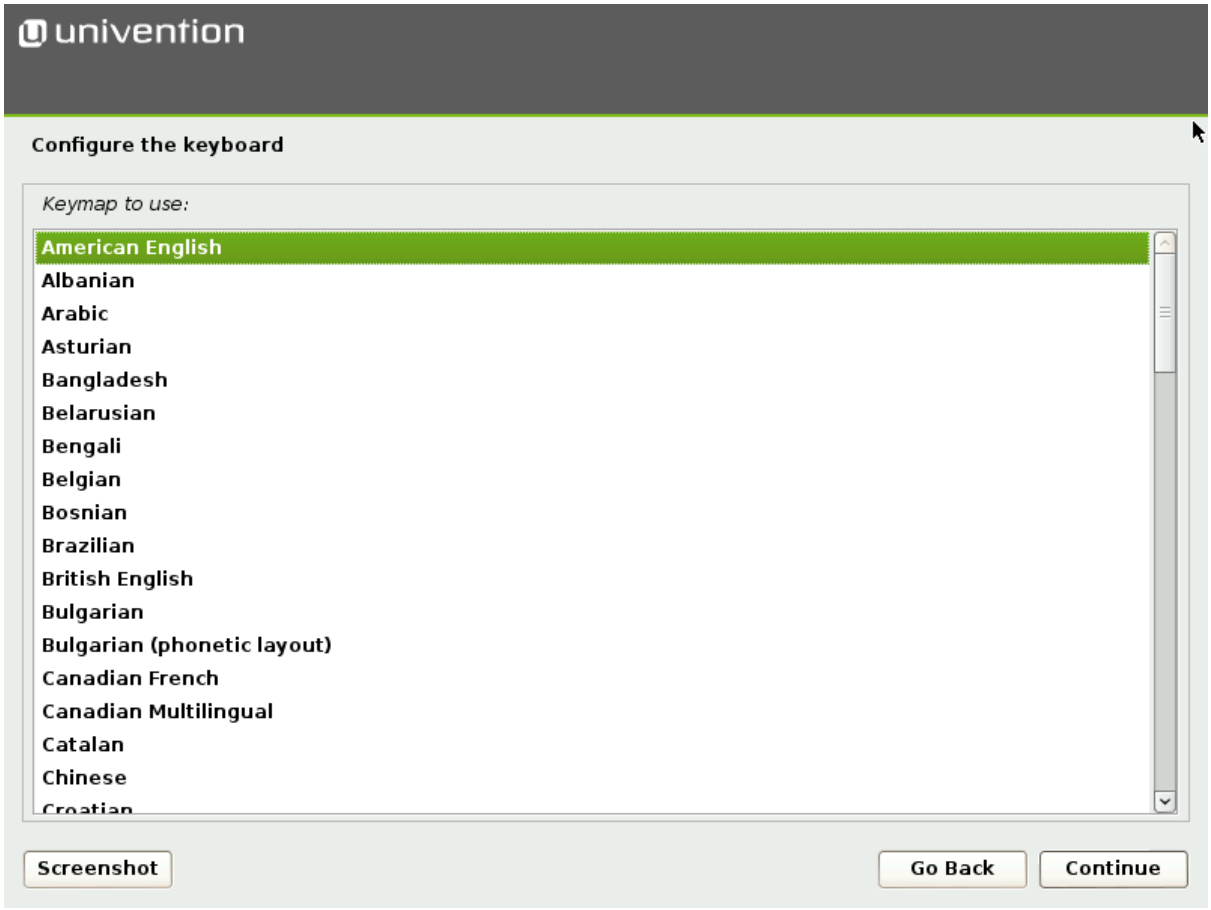


Fig. 2.4: Selecting the keyboard layout

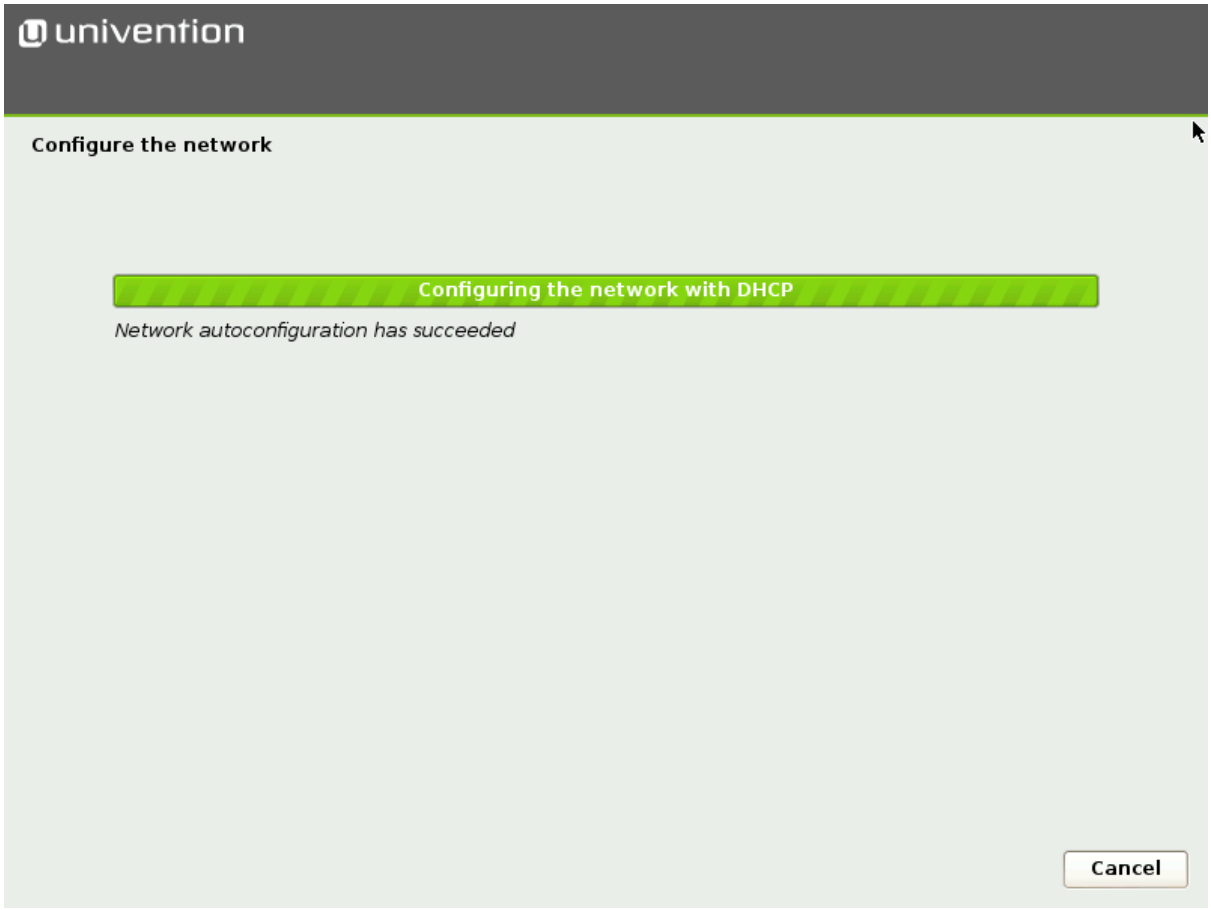


Fig. 2.5: Automatic network configuration

Note: At least one network interface is required for the installation of Univention Corporate Server. If no supported network card is detected, Univention Installer opens a list of supported drivers for selection.

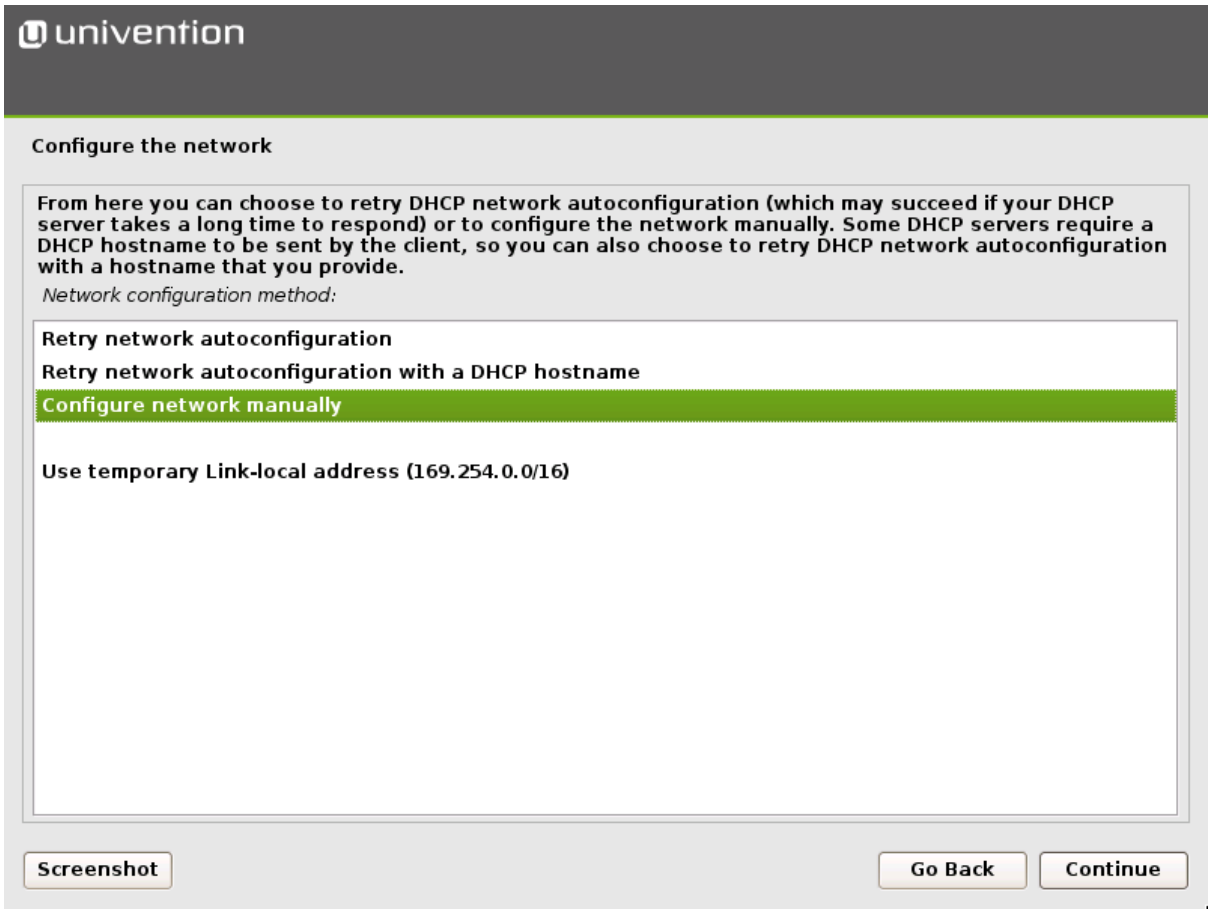


Fig. 2.6: Selecting the manual network configuration

In manual configuration it is possible to specify either a static IPv4 or an IPv6 address for the system. IPv4 addresses have a 32-bit length and are generally written in four blocks in decimal form (e.g., `192.0.2.10`), whereas IPv6 addresses are four times as long and typically written in hexadecimal form (e.g., `2001:0DB8:FE29:DE27:0000:0000:0000:0000`). In addition to entering a static IP address, values for network masks, gateways and DNS servers are also requested.

The following points must be taken into consideration when specifying a DNS server manually. They depend on the intended subsequent use of the UCS system.

- When installing the first UCS system in a new UCS domain, the IP address of the local router (if it provides the DNS service) or the DNS server of the internet provider should be entered.
- When installing every additional UCS system, the IP address of a UCS Directory Node system must be specified as the DNS server. This is essential for the automatic detection of the Primary Directory Node to function. In case of doubt, the IP address of the UCS Primary Directory Node system should be entered.
- If the UCS system is to join a Windows Active Directory domain during the installation, the IP address of an Active Directory domain controller system should be specified as the DNS server. This is essential for the automatic detection of the Windows Active Directory domain controller to function.

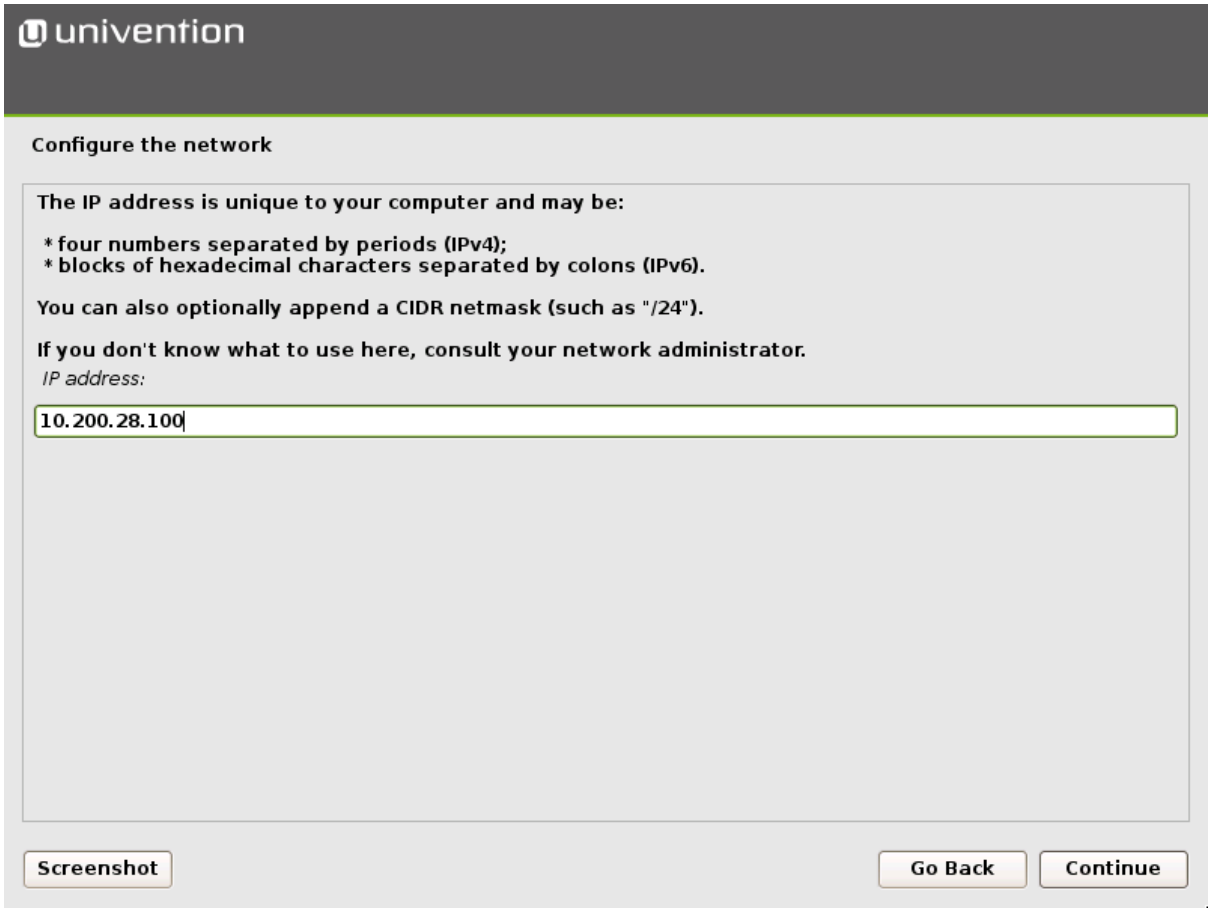


Fig. 2.7: Specifying an IP address

2.6 Setting up the root password

Setting of a password for the `root` user is required for logging on to the installed system. If a Primary Directory Node is installed, this password is also employed for the `Administrator` user. In later operation, the passwords for the `root` and `Administrator` users can be managed independently of each other. The password must be re-entered in the second entry field.

The password must contain at least eight characters for security reasons.

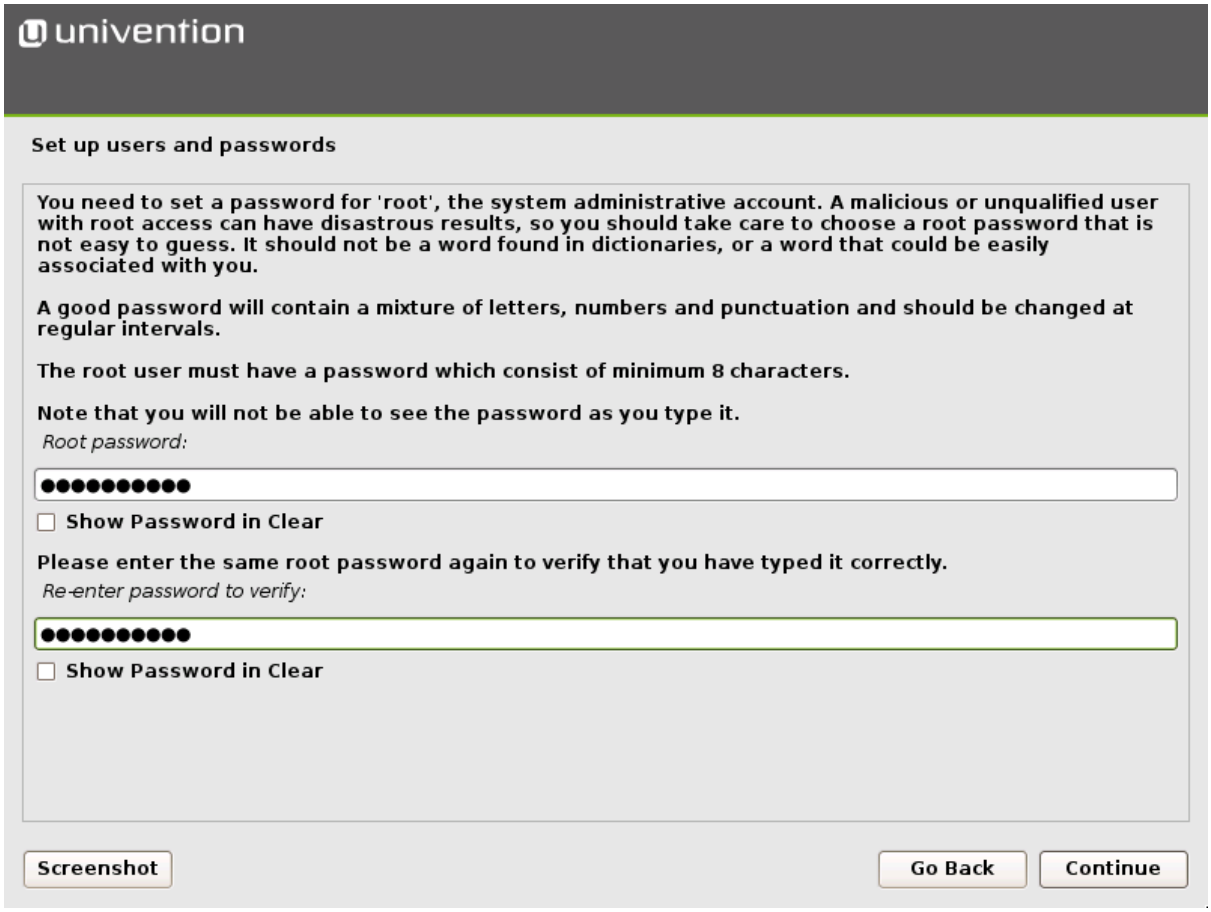


Fig. 2.8: Setting the root password

2.7 Partitioning the hard drive

The Univention Installer supports the partitioning of hard drives and the creation of different file systems (e.g., *ext4* and *XFS*). In addition, it is also possible to set up mechanisms such as the logical volume manager (LVM), RAID or partitions encrypted with LUKS.

As of UCS 4.0, the Univention Installer selects a suitable partition model (MBR or GPT) automatically depending on the size of the selected hard drive. On systems with the *Unified Extensible Firmware Interface (UEFI)*, the GUID Partition Table (GPT) is used automatically.

The Univention Installer offers guided installations to make installation simpler. In the guided installation, certain standard schemes with respect to the partitioning and formatting are applied to the selected hard drive. In addition, it is also possible to perform partitioning manually.

There are three schemes available for selection for guided partitioning:

Guided - Use entire disk
: In this scheme, an individual partition is created for each file system. Abstraction layers like LVM are not

used. During the following step the number of file systems or partitions is assigned. The size of the partitions is restricted to the size of the respective hard drive.

Guided - Use entire disk and set up LVM
If the second scheme is selected, a *Logical Volume Group (LVM)* is set up on the selected hard drive first. A separate logical volume is then created within the volume group for each file system. In this scheme, the size of the logical volume is restricted by the size of the volume group, which can also be subsequently enlarged with additional hard drives. In case of doubt, select this partitioning scheme.

Guided - Use entire disk with encrypted LVM
This version is the same as the previous version, with the addition that the LVM volume group is also encrypted. Consequently, the password for the encrypted volume group has to be entered every time the system is started up.

Caution: In all three versions, the existing data on the selected hard drive are deleted during the partitioning!

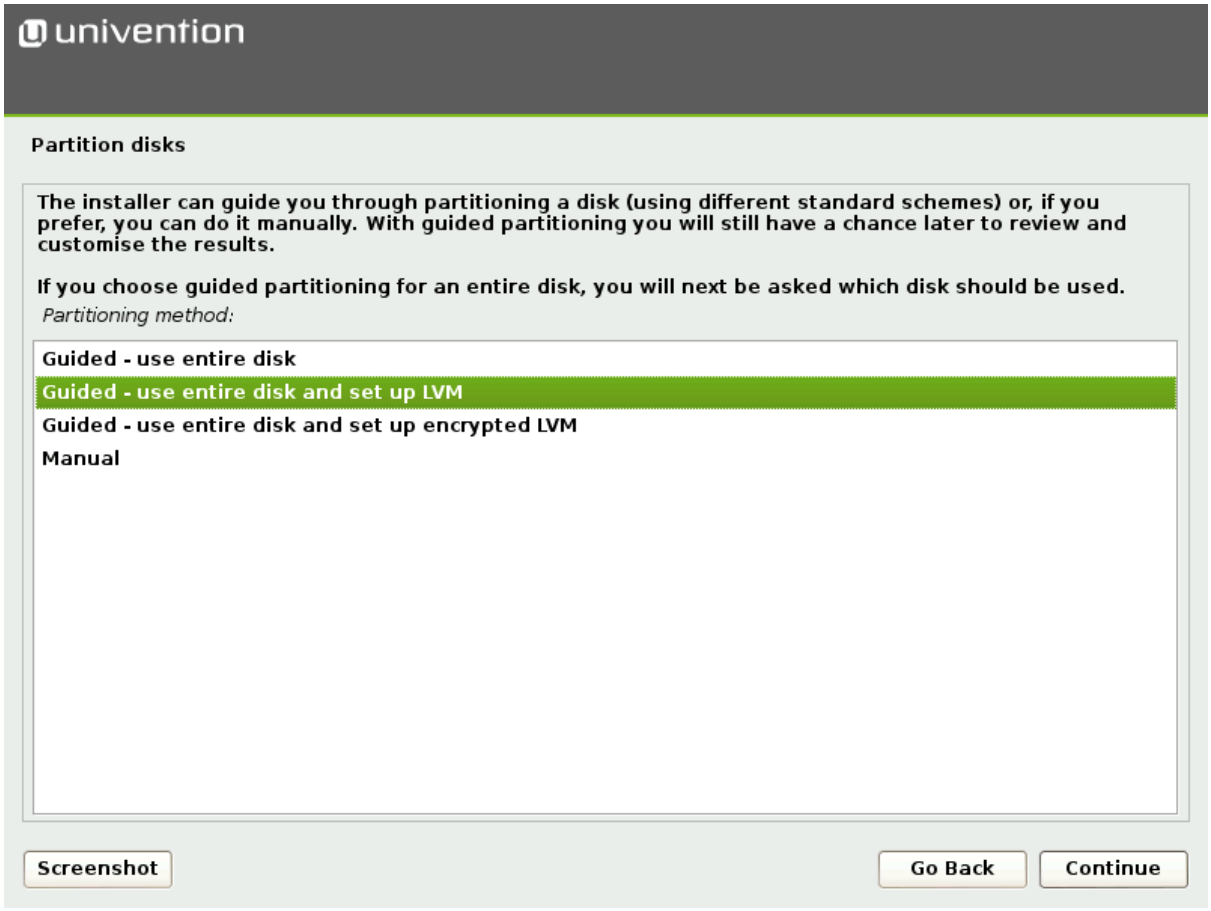


Fig. 2.9: Selecting the partitioning scheme

The next step is to select a hard drive from the list of those detected to which the partitioning version should be applied.

There are three sub versions for each partitioning version, which differ in the number of file systems created:

All files in one partition
In this version, just one partition or logical volume is created and the `/` file system saved there.

Separate `/home` partition
In addition to a file system for `/`, an additional file system is also created for `/home/`.

Separate `/home`, `/usr`, `/var` and `/tmp` partition

In addition to a file system for `/`, an additional file system is also created each for `/home/`, `/usr/`, `/var/` and `/tmp/`.

Before every active change to the hard drive, the change is displayed again in an additional dialogue and must be confirmed explicitly.



Fig. 2.10: Confirmation of changes to the hard drive

Once the partitioning is complete, the UCS basic system and additional software is installed automatically. This can take some time depending on the speed of the hardware used. The system is then made ready to boot via the installation of the GRUB boot loader.

A restart into the freshly installed system follows subsequently in order to complete the configuration within it.

2.8 Domain settings

The final configuration of the UCS system is started by selecting a domain mode. There are three modes available, which influence the following configuration steps:

Create a new UCS domain
In the first mode, *Create a new UCS domain*, the first system in a new UCS domain is configured: a UCS system with the Primary Directory Node system role. In the following configuration steps, the information required for setting up a new directory service, authentication service and DNS server are requested. A UCS domain can consist of one single or several UCS systems. Additional UCS systems can be added at a later point in time using the *Join an existing UCS domain* mode.

Join into an existing Active Directory domain
This mode, in which UCS is operated as a member of an Active Directory domain, is suitable for expanding an Active Directory domain with applications available on the UCS platform. Apps installed on the UCS

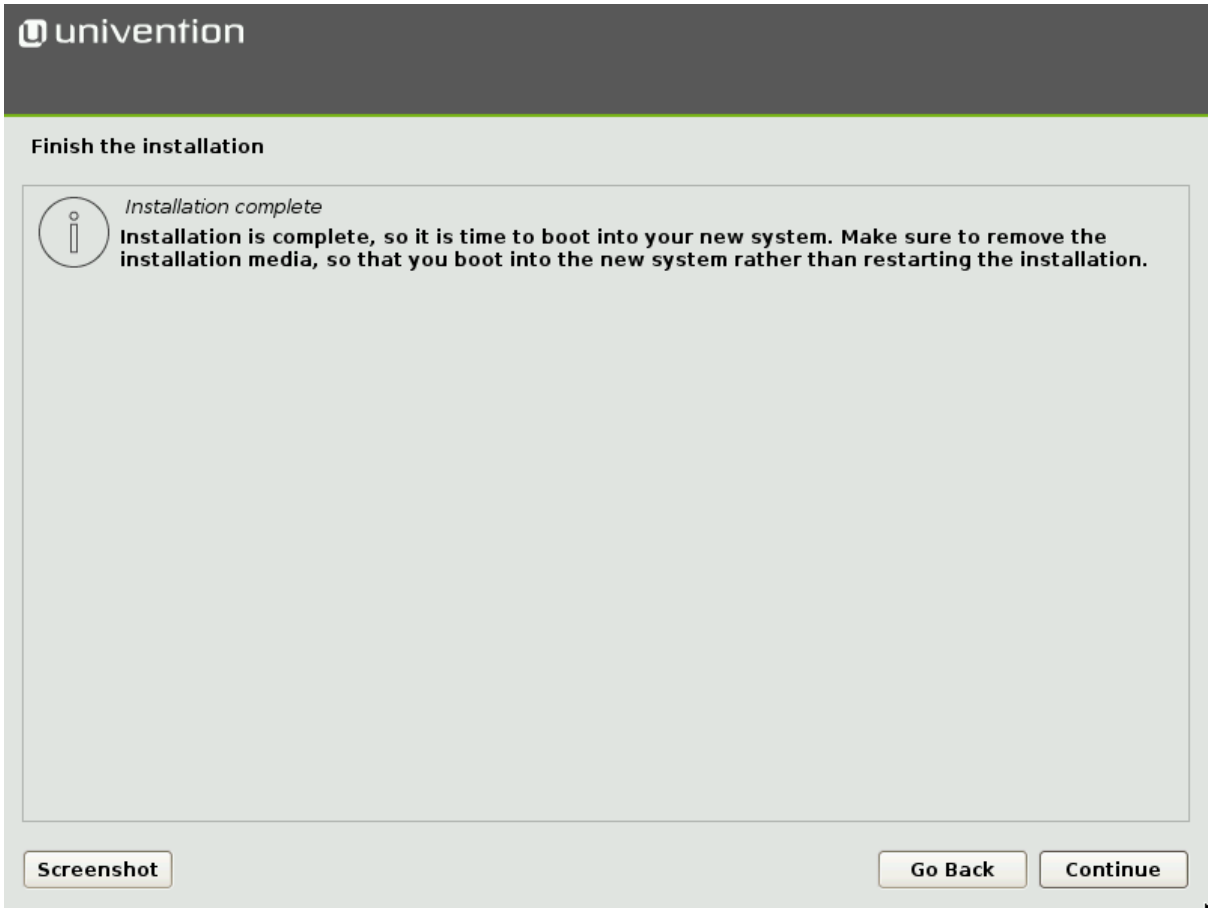


Fig. 2.11: Finish the installation

platform are then available for the users of the Active Directory domain to use. On selection of this mode, all the relevant information for the joining of the Active Directory domain is requested and the UCS system configured correspondingly.

Join into an existing UCS domain
Selecting the *Join into an existing UCS domain* mode allows the UCS system to be configured to join an existing UCS domain. What UCS system role it is to take on in the domain is queried at a later stage.

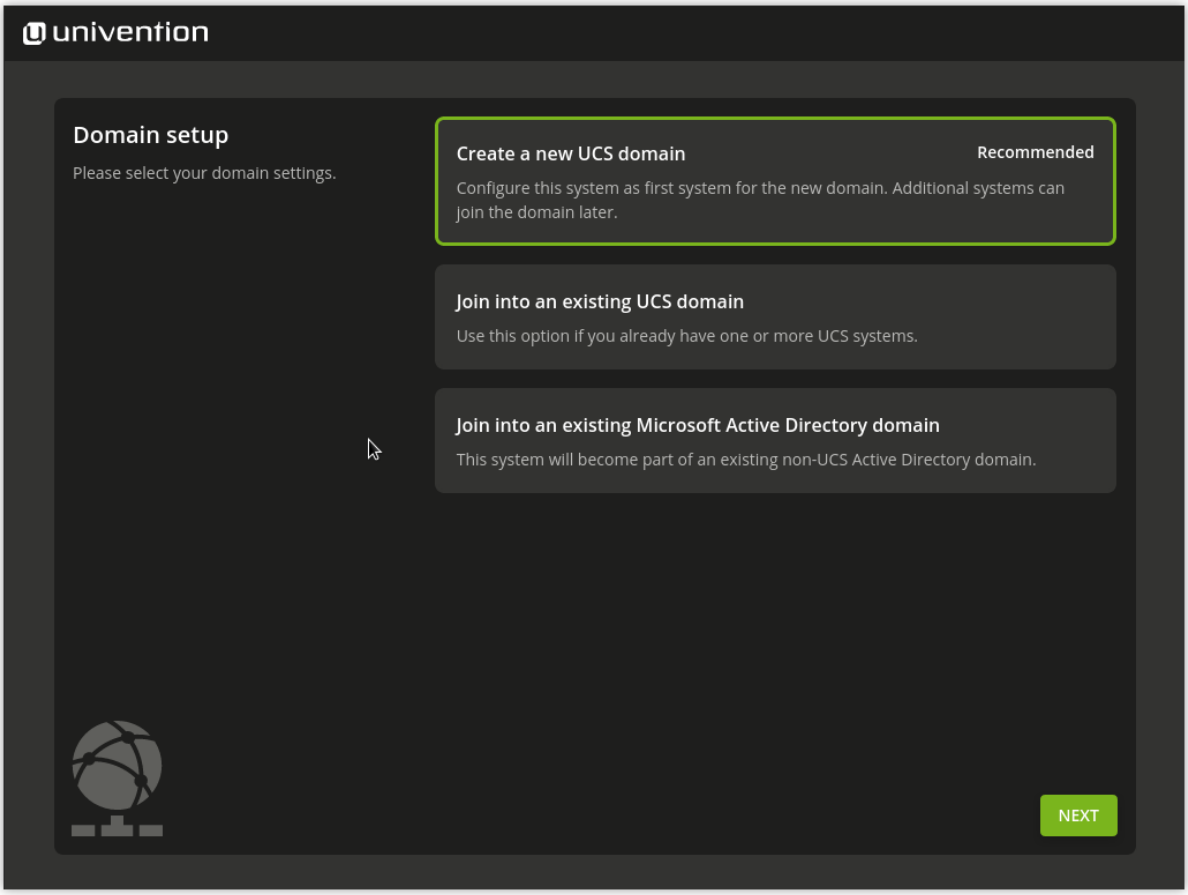


Fig. 2.12: Domain settings

2.8.1 Naming convention for hostnames

During UCS installation, the domain setup asks for a hostname and a domain name as *fully qualified domain name*. For compatibility reasons with Samba 4 and Windows domains, the hostname must adhere to the following naming convention:

- Length from 1 to 13 alpha numeric characters
- Only lower case letters (`a-z`) and numerals (`0-9`)
- Start and end with an alpha numeric character and can contain a hyphen (`-`) in between.

The naming convention has the following regular expression:

```
^[a-z0-9][a-z0-9-]{0,11}[a-z0-9]?$
```

2.8.2 *Create a new UCS domain* mode

Once the *Create a new UCS domain* mode has been selected, an *organization name*, an *email address*, a *fully qualified domain name* and an *LDAP base* are requested in the following two steps.

Specification of an organization name is optional and it is used in the second step to generate a domain name and the LDAP base automatically.

If a valid email address is specified, this is used to activate a personalized license, which is required for the use of the Univention App Center. The license is generated automatically and sent to the specified email address immediately. The license can then be imported via the UMC module *Welcome!* (*Activation of UCS license / license overview* (page 70)).

The name of the UCS system to be configured and the name of the DNS domain are determined from the fully qualified domain name (hostname including domain name) entered here. A suggestion is generated automatically from the organization name entered in the previous step. It is recommended not to use a publicly available DNS domain, as this can result in problems during the name resolution.

For the naming convention of the hostname, refer to *Naming convention for hostnames* (page 21).

A LDAP base needs to be specified for the initialization of the directory service. A suggestion is also derived here automatically from the fully qualified domain name. This value can usually be adopted without any changes.

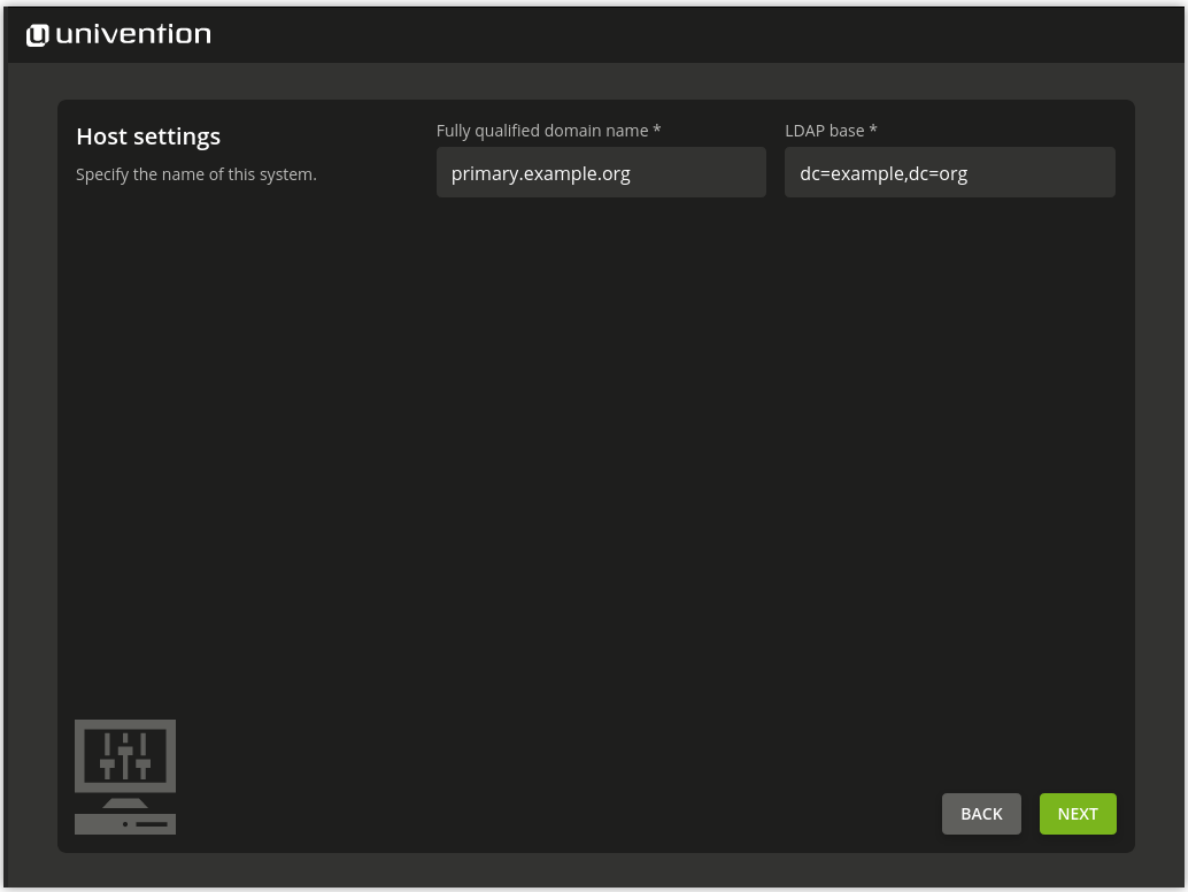


Fig. 2.13: Specification of hostname and LDAP base

2.8.3 *Join an existing Active Directory domain* mode

If the DNS server of an Active Directory domain was specified during the network configuration, the name of the Active Directory domain controller is suggested automatically in the *Active Directory account information* step. If the suggestion is incorrect, the name of another Active Directory domain controller or another Active Directory domain can be entered here.

The specification of an Active Directory account and the corresponding password is required for joining the Active Directory domain. The user account must possess the right to join new systems in the Active Directory domain.

In addition, a hostname must be entered for the UCS system to be configured. The suggested hostname can be adopted or a new hostname entered. The domain name of the computer is derived automatically from the domain DNS server. In some scenarios (e.g., a public mail server) it can prove necessary to use a specific fully qualified domain name. The UCS system will join the Active Directory domain with the hostname specified here. Once set up, the domain name **cannot** be changed again once the configuration is completed.

For the naming convention of the hostname, refer to *Naming convention for hostnames* (page 21).

In a UCS domain, systems can be installed in different *system roles*. The first UCS system, that joins an Active Directory domain, is automatically installed with the Primary Directory Node system role. If this mode is selected during installation of additional UCS systems, the system role selection dialogue is shown. The system roles are described within the following section.

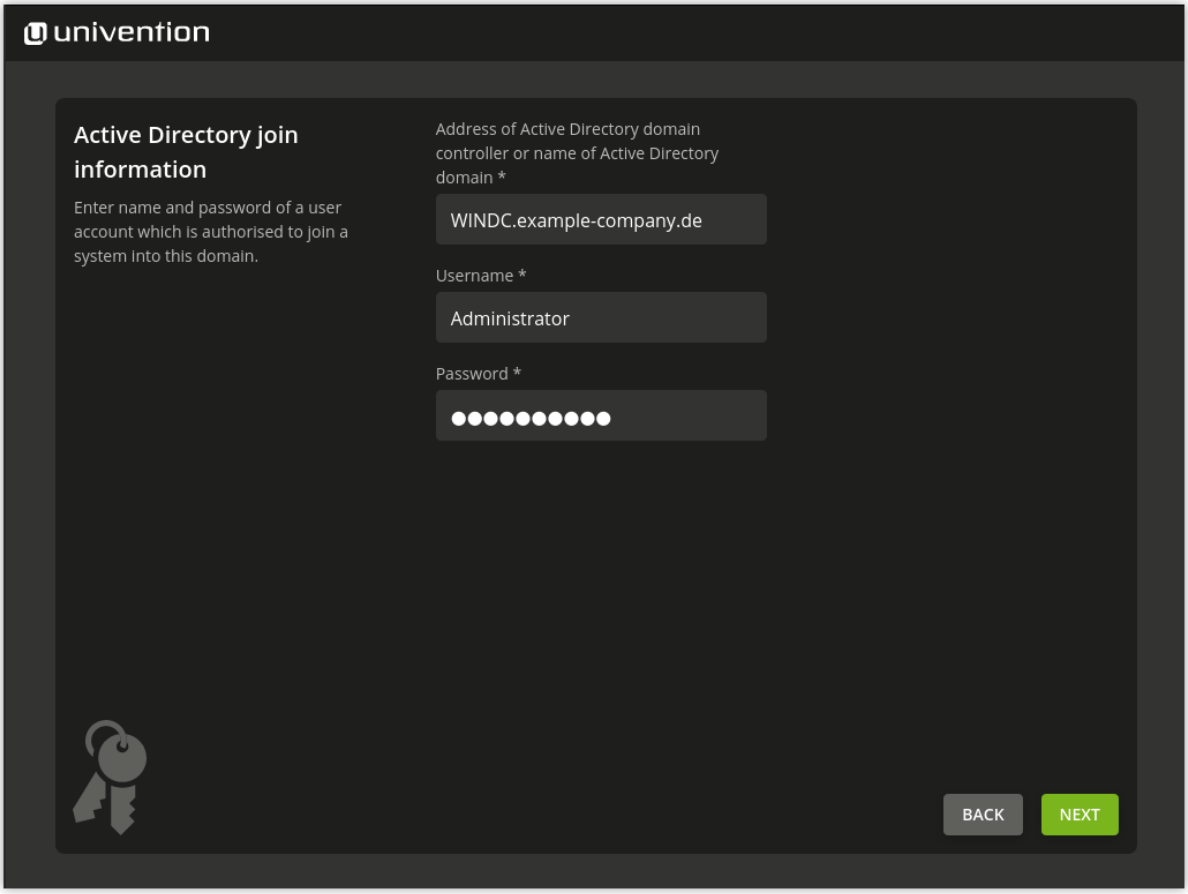


Fig. 2.14: Information on the Active directory domain

2.8.4 *Join an existing UCS domain domain* mode

In a UCS domain, systems can be installed in different *system roles*. The first system in a UCS domain is always installed with the Primary Directory Node system role. Additional UCS systems can join the domain at a later point in time and can be configured with one of the following system roles.

Backup Directory Node
: The Backup Directory Node is the fallback system for the Primary Directory Node. If the latter should fail, a Backup Directory Node can adopt the role of the Primary Directory Node permanently. All the domain data and SSL security certificates are saved as read-only copies on servers with the Backup Directory Node role.

Replica Directory Node
: All the domain data are saved as read-only copies on servers with the Replica Directory Node role. In contrast to the Backup Directory Node, however, not all security certificates are saved. As accesses to the services running on a Replica Directory Node are performed against the local LDAP directory service, Replica Directory Node systems are ideal for site servers and the distribution of high-load services.

Managed Node
: Managed Nodes are UCS systems without a local LDAP directory service. Access to domain data here is performed via other servers in the domain. They are therefore suitable for services which do not require a local database for authentication, for example, such as print and file servers.

Once the UCS system role has been selected, further information on the domain join is requested. If the domain join is not intended to occur automatically during the installation, the *Start join at the end of the installation* option can be disabled. If the correct DNS server was selected during the network configuration, Univention Installer can determine the name of the Primary Directory Node system automatically. If the decision is taken to join another UCS domain, the *Search Primary Directory Node in DNS* option can be disabled and the fully qualified domain name of the preferred Primary Directory Node entered in the input field below. The access information required for the domain join must be entered in the *Administrator account* and *Administrator password* input fields.

In addition, a hostname must be entered for the UCS system to be configured in the next step. The suggested hostname can be adopted or a new hostname entered. The domain name of the computer is derived automatically from the domain DNS server. In some scenarios (e.g., a public mail server) it can prove necessary to use a certain fully qualified domain name. Once set up, the domain name **cannot** be changed again once the configuration is completed.

For the naming convention of the hostname, refer to *Naming convention for hostnames* (page 21).

2.9 Confirming the settings

This dialogue shows the major settings that were made. If all the settings are correct, the *CONFIGURE SYSTEM* button can be used to start the configuration of the UCS system, see Fig. 2.16.

The *Update system after installation* option allows the automatic installation of available Errata updates. In addition, all patch level updates and Errata updates available are installed on a Primary Directory Node. On all other system roles, all the patch level updates are set up to the installation status of the Primary Directory Node. You need to sign in to the Primary Directory Node to check the installation status. This is done using the login data specified in the join options.

During the configuration, a progress bar displays the progress of the installation.

The installation protocol of the Univention Installer is saved in the following files:

- `/var/log/installer/syslog`
- `/var/log/univention/management-console-module-setup.log`

Completion of the configuration must be confirmed with the *CONFIGURE SYSTEM* button. The UCS system is then prepared for the first full booting procedure and restarted.

The system will then boot from the hard drive. Following the boot procedure, the `root` and `Administrator` users can sign in to the UCS portal page (see *UCS web interface* (page 55)), which can be reached under the IP address set during the installation or the hostname.

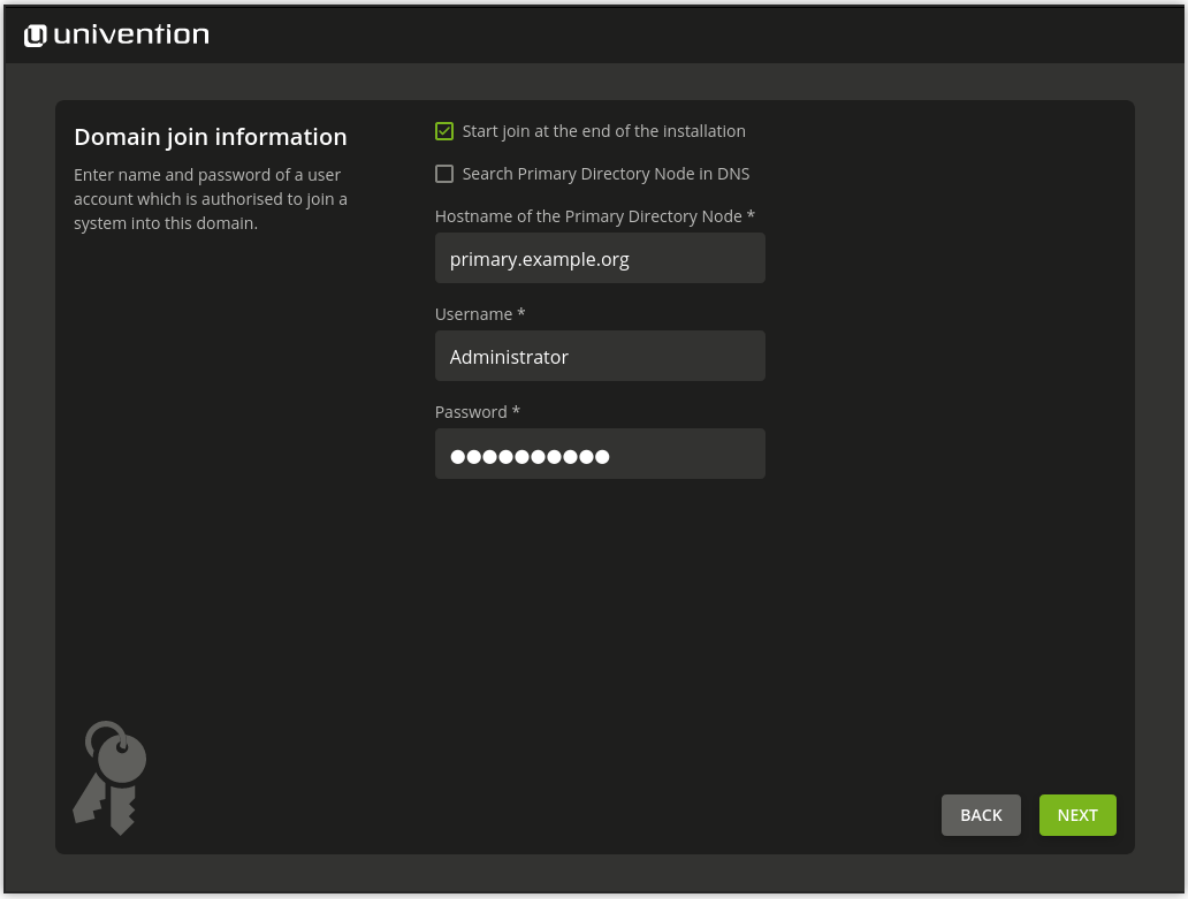


Fig. 2.15: Information on the domain join

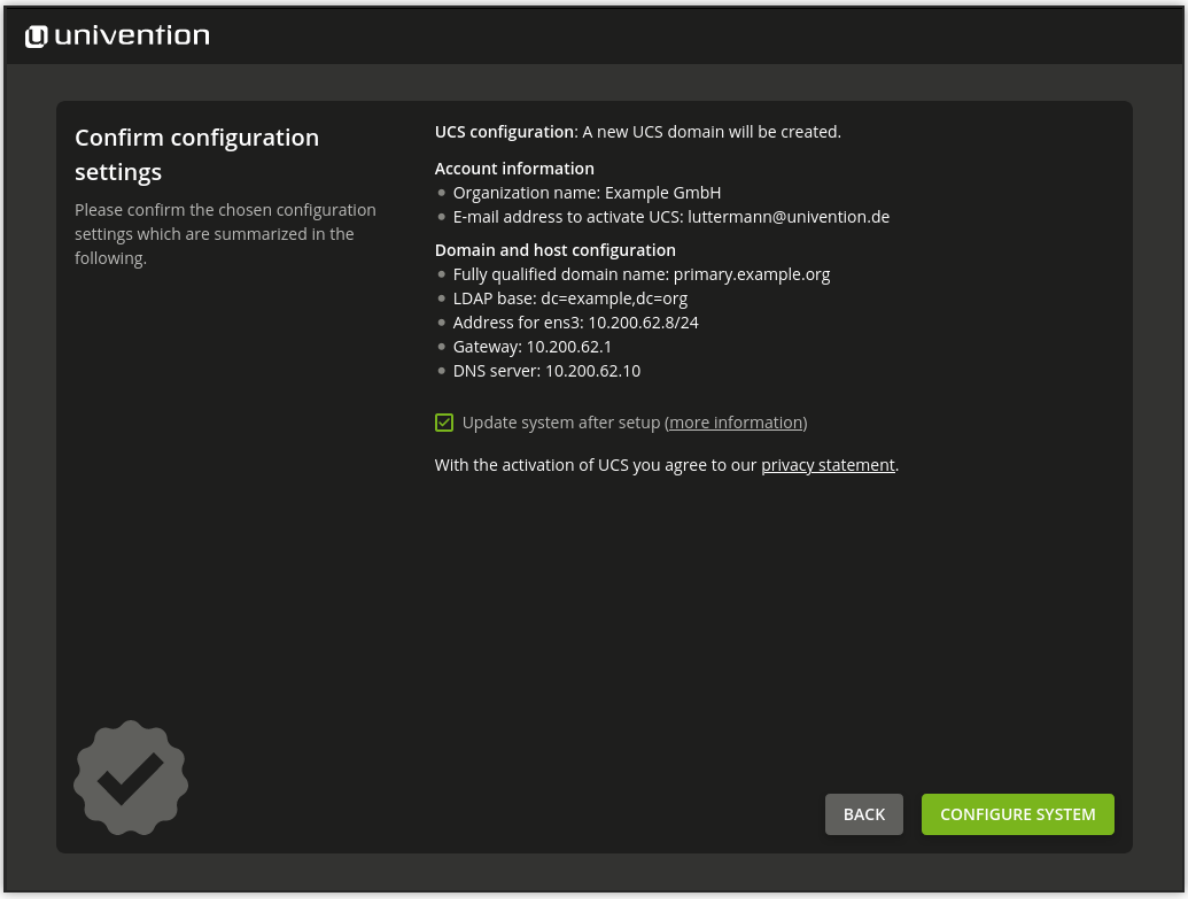


Fig. 2.16: Installation overview

If the computer was installed as the first system in the UCS domain (Primary Directory Node), the license can now be imported (see *Activation of UCS license / license overview* (page 70)).

2.10 Troubleshooting for installation problems

Information on possible installation problems can be found in the Univention Knowledge base[3] in the section *Installation*.

2.11 Installation in text mode

On systems that showed a problem with the graphic variant of Univention Installer, the installation may be also started in text mode. To achieve this, in the DVD boot menu *Advanced options* the entry *Install in text mode* has to be selected.

During installation in text mode Univention Installer shows the same information and asks for the same settings. After partitioning the hard drive, the system is prepared for the first boot and finally restarted.

After restart the configuration may be resumed by using a web browser. The URL `https://SERVER-IP-ADDRESS` or `http://SERVER-IP-ADDRESS` has to be opened within the browser (HTTPS is recommended). After loading the URL a login as user `root` is required.

The configuration process asks for location and network setting and then resumes with the same steps as the graphic variant of the installation, see *Domain settings* (page 19).

2.12 Installation in the Amazon EC2 cloud

Univention provides an Amazon Machine Image (AMI) for the Amazon EC2 cloud for UCS. This generic image for all UCS system roles is used to derive an individual instance which can be configured via Univention Management Console modules (domain name, software selection, etc.).

The process for setting up a UCS instance based on Amazon EC2 is documented in Univention Help 21833 - "Amazon EC2 Quickstart"[4].

2.13 Installation in VMware

If UCS is installed as a guest in VMware, select the option *Linux* ‣ *Debian* as the *Guest operating system*, because UCS is based on Debian.

The Linux kernel used in UCS includes all the support drivers necessary for operation in VMware (`vmw_balloon`, `vmw_pvsci`, `vmw_vmci`, `vmwgfx` and `vmxnet3`).

The open source version of the VMware Tools (Open VM Tools) is delivered with UCS. The tools can be installed using the `open-vm-tools` package (they are not required but do, for example, allow synchronization of the time on the virtualization server with the guest system).

[3] https://help.univention.com/c/knowledge-base/supported/48
[4] https://help.univention.com/t/21833

CHAPTER

THREE

DOMAIN SERVICES / LDAP DIRECTORY

Univention Corporate Server offers a cross platform domain concept with a common trust context between Linux and/or Windows systems. Within this domain a user is known to all systems via their username and password stored in the UCS Management System and can use all services which are authorized for them. The management system keeps the account synchronized for the windows login, Linux/POSIX systems and Kerberos. The management of user accounts is described in *User management* (page 105).

All UCS and Windows systems within a UCS domain have a host domain account. This allows system-to-system authentication. Domain joining is described in *Joining domains* (page 29).

The certificate authority (CA) of the UCS domain is operated on the Primary Directory Node. A SSL certificate is generated there for every system that has joined the domain. Further information can be found in *SSL certificate management* (page 43).

Every computer system which is a member of a UCS domain has a system role. This system role represents different permissions and restrictions, which are described in *UCS system roles* (page 33).

All domain-wide settings are stored in a directory service on the basis of OpenLDAP. *LDAP directory* (page 36) describes how to expand the managed attributes with LDAP scheme expansions, how to set up an audit-compliant LDAP documentation system and how to define access permissions to the LDAP directory.

Replication of the directory data within a UCS domain occurs via the Univention Directory Listener / Notifier mechanism. Further information can be found in *Listener/notifier domain replication* (page 40).

Kerberos is an authentication framework the purpose of which is to permit secure identification in the potentially insecure connections of decentralized networks. Every UCS domain operates its own Kerberos trust context (realm). Further information can be found in *Kerberos* (page 44).

3.1 Joining domains

A UCS, Ubuntu or Windows system must join the domain after installation.

In addition to UCS, Ubuntu and macOS, arbitrary Unix systems can be integrated into the domain. This is described in *Extended domain services documentation* [2].

3.1.1 How UCS systems join domains

There are three possibilities for a UCS system to join an existing domain:

- Directly after installation in the Univention Installer, see *Join an existing UCS domain domain mode* (page 24).
- Subsequently with the command **`univention-join`**, see *Subsequent domain joins with univention-join* (page 30).
- Using the UMC module *Domain join*, see *Joining domains via Univention Management Console module* (page 30).

The Primary Directory Node should always be installed at the most up-to-date release stand of the domains, as problems can arise with an outdated Primary Directory Node when a system using the current version joins.

When a computer joins, a computer account is created, the SSL certificates are synchronized and an LDAP copy is initiated if necessary. The *join scripts* are also run at the end of the join process. These register further objects, etc., in the directory service using the software packages installed on the system (see *Join scripts / Unjoin scripts* (page 31)).

The joining of the domain is registered on the client side in the `/var/log/univention/join.log` log file, which can be used for reference in error analysis. Actions run on the Primary Directory Node are stored in the `/home/<Join-Account>/.univention-server-join.log` log file.

The joining process can be repeated at any time. Systems may even be required to rejoin following certain administrative steps (such as changes to important system features on the Primary Directory Node).

Subsequent domain joins with *univention-join*

`univention-join` retrieves a number of essential parameters interactively; however, it can also be configured using a number of parameters:

`-dcname` `<HOSTNAME>`

The Primary Directory Node is usually detected via a DNS request. If that is not possible (e.g., a Replica Directory Node server with a different DNS domain is set to join), the computer name of the Primary Directory Node can also be entered directly using the `-dcname HOSTNAME` parameter. The computer name must then be entered as a fully qualified name, e.g., `primary.company.com`.

`-dcaccount` `<ACCOUNTNAME>`

A user account which is authorized to add systems to the UCS domains is called a join account. By default, this is the `Administrator` user or a member of the two groups `Domain Admins` and `DC Backup Hosts`. The join account can be assigned using the `-dcaccount ACCOUNTNAME` parameter.

`-dcpwd` `<FILE>`

The password can be set using the `-dcpwd FILE` parameter. The password is then read out of the specified file.

`-verbose`

The `-verbose` parameter is used to add additional debug output to the log files, which simplify the analysis in case of errors.

Joining domains via Univention Management Console module

A domain join can also be carried out web based via the UMC module *Domain join*. As the *Administrator* user does not yet exist on a system which has yet to join the domain, the login to the module is done as user `root`.

As for the *domain joining procedure via the command line* (page 30), username and password of a user account authorized to add computers to a domain must be entered in the resulting dialogue. Likewise, the Primary Directory Node will be determined automatically via a DNS request, but can also be entered manually.

The *Rejoin* option can be used to repeat the domain join at any time.

Join scripts / Unjoin scripts

Join scripts are run during the domain join. Examples for changes made by join scripts are the registration of a print server in the domain or the adaptation of DNS entries. Join scripts are components of the individual software packages. In the same way, there are also *unjoin scripts*, which can reset these changes following deinstallation of software components.

Join scripts are stored in the `/usr/lib/univention-install/` directory and unjoin scripts in `/usr/lib/univention-uninstall/`. Each join/unjoin script has a version. An example: A package has already been installed and the join script already run. The new version of the package now requires additional changes and the version number of the join script is increased.

The **`univention-check-join-status`** command can be used to check whether join/unjoin scripts need to be run (either because they have yet to be run or an older version was run).

Subsequent running of join scripts

If there are join/unjoin scripts on a system which have not yet been run or which can only be run for an older version, a warning message is shown upon opening a UMC module.

Join scripts that have not been run can be executed via the UMC module *Domain join* by clicking on the menu entry *Execute all pending join scripts*.

The **`univention-run-join-scripts`** command is used to run all of the join/unjoin scripts installed on a system. The scripts check automatically whether they have already been executed.

The name of the join/unjoin script and the output of the script are also recorded in `/var/log/univention/join.log`.

If **`univention-run-join-scripts`** is run on another system role than the Primary Directory Node, the user will be asked to input a username and password. This can be performed on the Primary Directory Node via the `--ask-pass` option.

3.1.2 Windows domain joins

The procedure for joining a Windows system to a UCS domain made available via Samba is now described as an example for Windows 11, Windows 10 and Windows 2012 / 2016 / 2019 / 2022. The process is similar for other Windows versions. In addition to the client versions, Windows server systems can also join the domain. Windows servers join the domain as member servers; joining a Windows systems as a domain controller is not supported. For more information about Windows in a UCS domain, refer to *Services for Windows* (page 171).

Only domain-compatible Windows versions can join the UCS domain, i.e., it is not possible for the Home versions of Windows to join a domain.

A host account is created for the Windows client automatically when it joins the domain (see *Management of computer accounts via Univention Management Console module* (page 143)). Information concerning MAC and IP addresses, the network, DHCP or DNS can be configured via UMC modules prior to or after joining the domain.

Domain joining is usually performed with the local Administrator account on the Windows system.

Joining the domain takes some time and the process must not be canceled prematurely. After successful joining a small window appears with the message *Welcome to the domain <your domain name>*. This should be confirmed with *OK*. The computer must then be restarted for the changes to take effect.

Domain names must be limited to 13 characters as they are otherwise truncated at the Windows client and this can lead to sign in errors.

For a domain join against a domain controller based on Samba/AD, the DNS configuration of the client must be set up in such a way that DNS entries from the DNS zone of the UCS domain can also be resolved. In addition, the time on the client system must also be synchronized with the time on the domain controller.

Windows 11

Joining a domain requires one of the editions *Pro*, *Education*, or *Enterprise* of Windows 11. To join Windows 11 into a UCS domain, use the following steps:

1. To open the Windows control panel, search for `Control Panel` in the field *Search*.
2. On the *Control Panel* navigate to *System and Security* ‣ *System* scroll down and click *Domain or workgroup*. Select *Change settings* ‣ *Change*.
3. Enable the option *Domain*.
4. Enter the name of the domain in the input field for the domain join. Use the full domain name, for example `mydomain.intranet`. Click the *OK* button.
5. Enter the *Username* and *Password* of a domain administrator account of the UCS domain in the respective input fields. In a UCS domain the default domain administrator username is `Administrator`.
6. Finally, to start the process for joining the domain, click *OK*.

Windows 10

The joining of domains is only possible with the Pro and Enterprise editions of Windows 10.

The control panel can be reached via the search field *Search the web and Windows*, which can be found in the start bar. Under *System and Security* ‣ *System* it must be clicked on *Change settings* ‣ *Change*.

The *Domain* option field must be ticked and the name of the domain must be entered in the input field for the domain join. The full domain name should be used, e.g. `mydomain.intranet`. After clicking on the *OK* button, the username of a domain administrator must be entered in the input field *Username*, by default this is `Administrator`. The password of the domain administrator has to be entered in the input field *Password*. Finally, the process for joining the domain can then be started by clicking on *OK*.

Windows Server 2012 / 2016 / 2019 / 2022

The control panel can be reached by moving the cursor to the bottom right-hand corner of the screen. The *Control Panel* can then be searched for under *Search* ‣ *Apps*. *Change settings* ‣ *Network ID* must be clicked on under *System and Security* ‣ *System*.

The *Domain* option field must be ticked and the name of the Samba domain entered in the input field for the domain join. After clicking on the *OK* button, the username `Administrator` must be entered in the input field *Name* and the password from `uid=Administrator,cn=users,`*`LDAP base DN`* transferred to the *Password* input field. The process for joining the domain can then be started by clicking on *OK*.

3.1.3 Ubuntu domain joins

Univention provides the **`Univention Domain Join Assistant`** to integrate Ubuntu clients into a UCS domain. Documentation and installation instructions are available at Github[5].

[5] https://github.com/univention/univention-domain-join

3.1.4 macOS domain joins

UCS supports domain joins of macOS clients into a UCS environment using Samba/AD. This documentation refers to macOS 10.8.2.

The domain join can be performed using the system preferences menu or the **dsconfigad** command line tool.

After the domain join it is possible to automatically mount CIFS shares to subfolders in `/Volumes` when logging in with a domain user. For that, the following line has to be added to the file `/etc/auto_master`:

```
/Volumes      auto_custom
```

In addition, the file `/etc/auto_custom` needs to be created and the shares which should be mounted have to be listed in it in the following way:

```
<SUBFOLDER_NAME>    -fstype=smbfs    ://<FQDN>/<SHARE_NAME>
```

Note that the automatically mounted shares are not displayed in the finder's sidebar.

Domain join using the system preferences GUI

In the System Preferences via the *Users & Groups* entry, the *Login menu* can be reached. After authenticating by clicking on the lock in the lower left corner and providing credentials of a local *Administrator* account, the *Network Account Server: Join* button needs to be clicked. From that menu it is possible to open the *Directory Utility*.

In the advanced options section, the option *Create mobile account at login* should be activated. A mobile account has the advantage that, when the domain is not available, the user can log into the macOS system with the same account used for logging into the domain.

After filling in the domain name in the field *Active Directory Domain* and the hostname of the macOS client in the field *Computer ID*, the join process is initiated after clicking the button *Bind...*. The username and password of an account in the `Domain Admins` group needs to be entered, e.g., `Administrator`.

Domain join on the command line

The domain join can also be performed on the command line using **dsconfigad**:

```
$ dsconfigad -a <MAC HOSTNAME> \
  -domain <FQDN> \
  -ou "CN=Computers,<LDAP base DN>" \
  -u <Domain Administrator> \
  -mobile enable
```

Additional configuration options are available through **dsconfigad -help**.

3.2 UCS system roles

In a UCS domain systems can be installed in different *system roles*. The following gives a short characterization of the different systems.

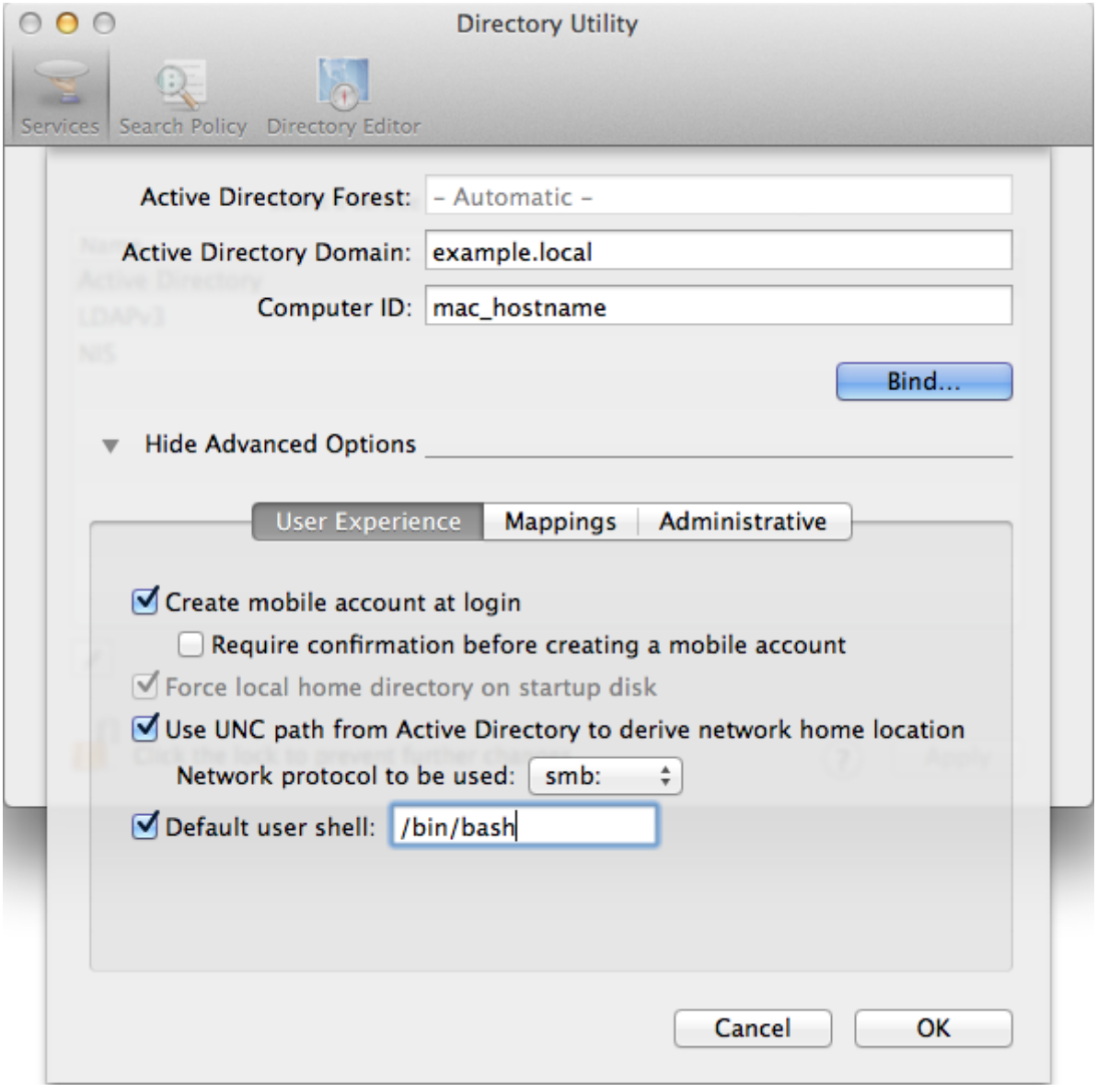


Fig. 3.1: Domain join of a macOS system

3.2.1 Primary Directory Node

A system with the Primary Directory Node role is the primary domain controller of a UCS domain and is always installed as the first system. The domain data (such as users, groups, printers) and the SSL security certificates are saved on the Primary Directory Node.

Copies of these data are automatically transferred to all servers with the Backup Directory Node role.

3.2.2 Backup Directory Node

All the domain data and SSL security certificates are saved as read-only copies on servers with the Backup Directory Node role.

The Backup Directory Node is the fallback system for the Primary Directory Node. If the latter should fail, a Backup Directory Node can take over the role of the Primary Directory Node permanently (see *Converting a Backup Directory Node backup to the new Primary Directory Node* (page 49)).

3.2.3 Replica Directory Node

All the domain data are saved as read-only copies on servers with the Replica Directory Node role. In contrast to the Backup Directory Node, however, not all security certificates are synchronized.

As access to the services running on a Replica Directory Node are performed against the local LDAP server, Replica Directory Nodes are ideal for site servers and the distribution of load-intensive services.

A Replica Directory Node cannot be promoted to a Primary Directory Node.

3.2.4 Managed Node

Managed Node are server systems without a local LDAP server. Access to domain data here is performed via other servers in the domain.

3.2.5 Ubuntu

Ubuntu clients can be managed with this system role, see *Integration of Ubuntu clients* (page 148).

3.2.6 Linux

This system role is used for the integration of other Linux systems than UCS and Ubuntu, e.g., for Debian or CentOS systems. The integration is documented in *Extended domain services documentation* [2].

3.2.7 macOS

macOS systems can be joined into a UCS domain using Samba/AD. Additional information can be found in *macOS domain joins* (page 33).

3.2.8 Domain Trust Account

A domain trust account is set up for trust relationships between Windows and UCS domains.

3.2.9 IP client

An IP client allows the integration of non-UCS systems into the IP management (DNS/DHCP), e.g., for network printers or routers.

3.2.10 Windows Domaincontroller

Windows domain controllers in a Samba/AD environment are operated with this system role.

3.2.11 Windows Workstation/Server

Windows clients and Windows Managed Nodes are managed with this system role.

3.3 LDAP directory

Univention Corporate Server saves domain-wide data in a LDAP directory service based on OpenLDAP. This chapter describes the advanced configuration and coordination of OpenLDAP.

Often several LDAP servers are operated in a UCS domain. The configuration of the server(s) used is described in *Configuration of the LDAP server in use* (page 165).

Note: The LDAP directory is part of Univention Nubus in the *Identity Store and Directory Service* component. For more information about Nubus, refer to *What is Univention Nubus?* (page 2)

3.3.1 LDAP schemas

Schema definitions specify which object classes exist and which attributes they include, i.e., which data can be stored in a directory service. Schema definitions are saved as text files and included in the OpenLDAP server's configuration file.

UCS uses standard schemas where possible in order to allow interoperability with other LDAP applications. Schema extensions are supplied for Univention-specific attributes - such as for the policy mechanism.

LDAP schema extensions

To keep the efforts required for small extensions in LDAP as low as possible, Univention Corporate Server provides its own LDAP scheme for customer extensions. The LDAP object class `univentionFreeAttributes` can be used for extended attributes without restrictions. It offers 20 freely usable attributes (`univentionFreeAttribute1` to `univentionFreeAttribute20`) and can be used in connection with any LDAP object (e.g., a user object).

If LDAP schema extensions are to be delivered as part of software packages, there is also the possibility of packaging them and distributing them to all the Backup Directory Node servers in the domain using a Univention Directory Listener module. Further information is available in Packaging LDAP Schema Extensions[6].

[6] https://docs.software-univention.de/developer-reference/5.0/en/ldap.html#settings-ldapschema

LDAP schema replication

The replication of the LDAP schemas is also automated via the listener/notifier mechanism (see *Listener/notifier domain replication* (page 40)). This relieves the administrator of the need to perform all schema updates manually on all the OpenLDAP servers in the domain. Performing the schema replication before the replication of LDAP objects guarantees that this doesn't fail as a result of missing object classes or attributes.

On the Primary Directory Node, a checksum for all the directories with schema definitions is performed when the OpenLDAP server is started. This checksum is compared with the last saved checksum in the `/var/lib/univention-ldap/schema/md5` file.

The actual replication of the schema definitions is initiated by the Univention Directory Listener. Prior to every request from the Univention Directory Notifier for a new transaction ID, its current schema ID is requested. If this is higher than the schema ID on the listener side, the currently used sub-schema is procured from the notifier system's LDAP server via an LDAP search.

The output sub-schema is included on the listener system in LDIF format in the `/var/lib/univention-ldap/schema.conf` file and the local OpenLDAP server restarted. If the schema replication is completed with this step, the replication of the LDAP objects is continued.

3.3.2 Audit-proof logging of LDAP changes

The **`univention-directory-logger`** package allows the logging of all changes in the LDAP directory service. As each data record contains the hash value of the previous data record, manipulations of the log file - such as deleted entries - can be uncovered.

To install the **`univention-directory-logger`** package, follow the instructions for installing software packages on UCS in *Installation/removal of individual packages via Univention Management Console module* (page 100) or *Installation/removal of individual packages in the command line* (page 101).

Individual areas of the directory service can be excluded from the logging. These branches can be configured using the Univention Configuration Registry variables *ldap/logging/exclude1* (page 296), *ldap/logging/excludeN* (page 296), etc. As standard, the container is excluded in which the temporary objects are stored (`cn=temporary,cn=univention`). The LDAP changes are logged by a Univention Directory Listener module. The Univention Directory Listener service must be restarted if changes are made to the Univention Configuration Registry variables.

The logging is made in the `/var/log/univention/directory-logger.log` file in the following format:

```
START
Old Hash: Hash sum of the previous data record
DN: DN of the LDAP object
ID: Listener/notifier transaction ID
Modifier: DN of the modifying account
Timestamp: Time stamp in format dd.mm.yyyy hh:mm:ss
Action: add, modify or delete

Old Values:
 List of old attributes, empty when an object is added
New Values:
 List of new attributes, empty when an object is deleted
END
```

A hash sum is calculated for each logged data record and also logged in the `daemon.info` section of the syslog service.

As of UCS 4.4 erratum 536[7] the respective transaction ID of the entry is added to the file `/var/log/univention/directory-logger.log` before each line as a prefix:

[7] https://errata.software-univention.de/#/?erratum=4.4x536

```
ID 342: START
ID 342: Old Hash: 70069d51a7e2e168d7c7defd19349985
ID 342: DN: uid=Administrator,cn=users,dc=example,dc=com
ID 342: ID: 342
ID 342: Modifier: cn=admin,dc=example,dc=com
ID 342: Timestamp: 15.04.2020 09:20:40
ID 342: Action: modify
ID 342:
ID 342: Old values:
ID 342: description: This is a description test
ID 342: entryCSN: 20200415091936.317108Z#000000#000#000000
ID 342: modifyTimestamp: 20200415091936Z
ID 342:
ID 342: New values:
ID 342: description: This is a description test
ID 342: entryCSN: 20200415092040.430976Z#000000#000#000000
ID 342: modifyTimestamp: 20200415092040Z
ID 342: END
```

If **univention-directory-logger** was installed before this UCS version, the old behavior (no prefix) is retained by default. By setting the Univention Configuration Registry Variable *ldap/logging/id-prefix* (page 296) to `yes` the new behavior can be activated. This prefix simplifies the correlation of related lines when post-processing the sign in analysis and monitoring software.

3.3.3 Timeout for inactive LDAP connections

The Univention Configuration Registry Variable *ldap/idletimeout* (page 296) is used to configure a time period in seconds after which the LDAP connection is cut off on the server side. When the value is set to `0`, no expiry period is in use. The timeout period has been set at six minutes as standard.

3.3.4 LDAP command line tools

In addition to the UMC web interface, there are also a range of programs with which one can access the LDAP directory from the command line.

The **univention-ldapsearch** tool simplifies the authenticated search in the LDAP directory. A search filter needs to be specified as an argument; in the following example, the administrator is searched for using the user ID:

```
$ univention-ldapsearch uid=Administrator
```

The **slapcat** command makes it possible to save the current LDAP data in a text file in LDIF format, e.g.:

```
$ slapcat -f /etc/ldap/slapd.conf > ldapdata.txt
```

3.3.5 Access control for the LDAP directory

Access to the information contained in the LDAP directory is controlled by Access Control Lists (ACLs) on the server side. The ACLs are defined in the central configuration file `/etc/ldap/slapd.conf` and managed using Univention Configuration Registry.

The `slapd.conf` is managed using a multifile template; further ACL elements can be added below `/etc/univention/templates/files/etc/ldap/slapd.conf.d/` between the `60univention-ldap-server_acl-master` and `70univention-ldap-server_acl-master-end` files or the existing templates expanded upon.

If LDAP ACL extensions are to be delivered as part of software packages, there is also the possibility of packaging them and distributing them to all the LDAP servers in the domain using a Univention Directory Listener module. Further information is available in Packaging LDAP ACL Extensions[8].

The default setting of the LDAP server after new installations with UCS does not allow anonymous access to the LDAP directory. This behavior is configured with the Univention Configuration Registry Variable *ldap/acl/read/anonymous* (page 295). Individual IP addresses can be granted anonymous read permissions via Univention Configuration Registry Variable *ldap/acl/read/ips* (page 295).

Following successful authentication on the LDAP server, all attributes of a user account can be read out by this user.

In addition, an extra, internal account, the root DN, also has full write access.

In addition, UCS offers a number of further ACLs installed as standard which suppress access to sensitive files (e.g., the user password) and establish rules which are necessary for operation (e.g., necessary accesses to computer accounts for log-ins). The read and write access to this sensitive information if only intended for members of the `Domain Admins` group.

Nested groups are also supported. The Univention Configuration Registry Variable *ldap/acl/nestedgroups* (page 296) can be used to deactivate the nested groups function for LDAP ACLs, which will result in a speed increase for directory requests.

Delegation of the privilege to reset user passwords

To facilitate the delegation of the privilege to reset user passwords, the **univention-admingrp-user-passwordreset** package can be installed. It uses a join script to create the `User Password Admins` user group, in so far as this does not already exist.

Members of this group receive the permission via additional LDAP ACLs to reset the passwords of other users. These LDAP ACLs are activated automatically during the package installation. To use another group, or a group that already exists, instead of the `User Password Admins` group, the DN of the group to be used can be entered in the Univention Configuration Registry Variable *ldap/acl/user/passwordreset/accesslist/groups/dn* (page 296). The LDAP server must be restarted after making changes.

Passwords can be reset via the UMC module *Users*. By default the module is only accessible to the `Administrator` user. During the installation a new `default-user-password-admins` policy is created automatically, which is linked to the members of the `User Password Admins` group and can be assigned to a corresponding container in the LDAP directory. Further information on the configuration of UMC policies can be found in *Delegated administration for UMC modules* (page 81).

The policy makes it possible to search for users and create an overview of all the attributes of a user object. If an attempt is made to modify further attributes in addition to the password when the user does not have sufficient access rights to the LDAP directory, Univention Directory Manager denies them write access with the message *Permission denied*.

Caution: The package should be installed on the Primary Directory Node and the Backup Directory Nodes. During the installation, the LDAP server is restarted and is thus temporarily unavailable.

Password resets via the password group can be prevented for sensitive users or groups (e.g., domain administrators). The Univention Configuration Registry variables *ldap/acl/user/passwordreset/protected/uid* (page 296) and *ldap/acl/user/passwordreset/protected/gid* (page 296) can be used to configure users and groups. Multiple values must be separated by commas. After changes to the variables, it is necessary to restart the LDAP server using the **systemctl restart slapd** command. By default the members of the `Domain Admins` group are protected against having theirs password changed.

If access to additional LDAP attributes should be necessary for changing the password, the attribute names can be expanded in Univention Configuration Registry Variable *ldap/acl/user/passwordreset/attributes* (page 296). After the change, the LDAP directory service must be restarted for the change to take effect. This variable is already set appropriately for a UCS standard installation.

[8] https://docs.software-univention.de/developer-reference/5.0/en/ldap.html#settings-ldapacl

3.3.6 Name Service Switch / LDAP NSS module

With the *Name Service Switch*, the GNU C standard library (**glibc**) used in Univention Corporate Server offers a modular interface for resolving the names of users, groups and hosts.

The LDAP NSS module is used on UCS systems for access to the domain data (e.g., users) as standard. The module queries the LDAP server specified in the Univention Configuration Registry Variable *ldap/server/name* (page 297) (and if necessary the *ldap/server/addition* (page 297)).

What measures should be taken if the LDAP server cannot be reached can be specified by the Univention Configuration Registry Variable *nssldap/bindpolicy* (page 301). As standard, if the server cannot be reached, a new connection attempt is made. If the variable is set to `soft`, then no new attempt is made to connect. This can considerably accelerate the boot of a system if the LDAP server cannot be reached, e.g., in an isolated test environment.

3.3.7 Syncrepl for synchronization with non-UCS OpenLDAP servers

The syncrepl replication service can also be activated parallel to the notifier service for the synchronization of OpenLDAP servers not installed on UCS systems. Syncrepl is a component of OpenLDAP, monitors changes in the local directory service and transmits them to other OpenLDAP servers.

3.3.8 Configuration of the directory service when using Samba/AD

As standard, the OpenLDAP server is configured in such a way that it also accepts requests from ports `7389` and `7636` in addition to the standard ports `389` and `636`.

If Samba/AD is used, the Samba/AD domain controller service occupies the ports `389` and `636`. In this case, OpenLDAP is automatically reconfigured so that only ports `7389` and `7636` are used. This must be taken into account during the configuration of syncrepl in particular (see *Syncrepl for synchronization with non-UCS OpenLDAP servers* (page 40)). **univention-ldapsearch** uses the standard port automatically.

3.3.9 Daily backup of LDAP data

The content of the LDAP directory is backed up daily on the Primary Directory Node and all Backup Directory Node systems via a Cron job. If Samba 4 is used, its data directory is also backed up.

The LDAP data are stored in the `/var/univention-backup/` directory in the naming scheme `ldap-backup_DATE.ldif.gz` in LDIF format. They can only be read by the `root` user. The Samba 4 files are stored in the directory `/var/univention-backup/samba/`.

The Univention Configuration Registry Variable *backup/clean/max_age* (page 291) can be used to define how long old backup files are kept (e.g. *backup/clean/max_age* (page 291)=`365`, all files older than `365` days are automatically deleted). For new installations (from UCS 4.4-7 on) the default for this variable is `365` (days). If the variable is not set, no backup files are deleted.

3.4 Listener/notifier domain replication

3.4.1 Listener/notifier replication workflow

Replication of the directory data within a UCS domain occurs via the Univention Directory Listener/Notifier mechanism:

- The Univention Directory Listener service runs on all UCS systems.
- On the Primary Directory Node (and possibly existing Backup Directory Node systems) the Univention Directory Notifier service monitors changes in the LDAP directory and makes the selected changes available to the Univention Directory Listener services on the other UCS systems.

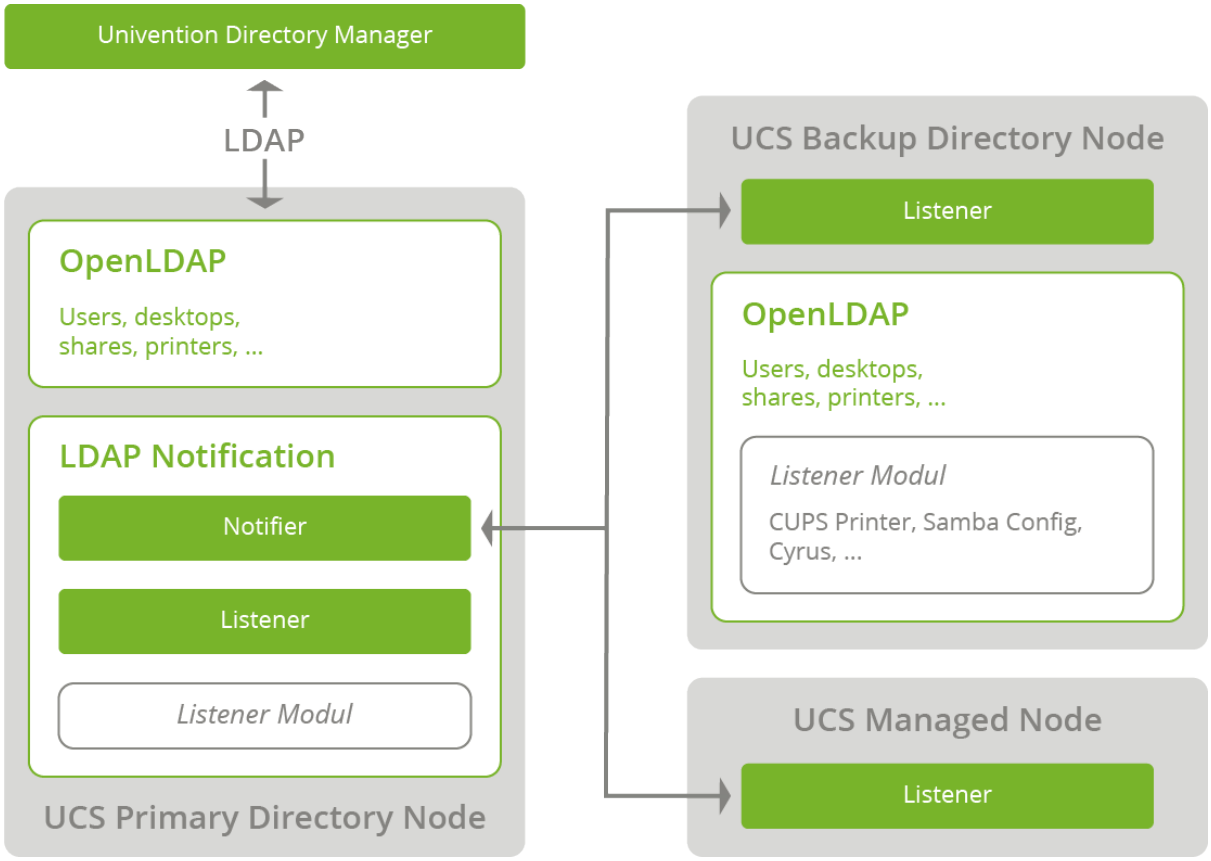


Fig. 3.2: Listener/Notifier mechanism

The active Univention Directory Listener instances in the domain connect to a Univention Directory Notifier service. If an LDAP change is performed on the Primary Directory Node (all other LDAP servers in the domain are read-only), this is registered by the Univention Directory Notifier and notified to the listener instances.

Each Univention Directory Listener instance uses a range of Univention Directory Listener modules. These modules are shipped by the installed applications; the print server package includes, for example, listener modules which generate the CUPS configuration.

Univention Directory Listener modules can be used to communicate domain changes to services which are not LDAP-compatible. The print server CUPS is an example of this: The printer definitions are not read from the LDAP, but instead from the `/etc/cups/printers.conf` file. Now, if a printer is saved in the UMC printer management, it is stored in the LDAP directory. This change is detected by the Univention Directory Listener module *cups-printers* and an entry added to, modified or deleted in `/etc/cups/printers.conf` based on the data in the LDAP.

Additional information on the setup of Univention Directory Listener modules and developing your own modules can be found in *Univention Developer Reference* [3].

LDAP replication is also performed by a listener module. If the LDAP server to be replicated to is not accessible, the LDAP changes are temporarily stored in the `/var/lib/univention-directory-replication/failed.ldif` file. The contents of the file are automatically transferred to the LDAP when the LDAP server is available again.

The listener/notifier mechanism works based on transactions. A transaction ID is increased for every change in the LDAP directory of the Primary Directory Node. A Univention Directory Listener instance which has missed several transactions - for example, because the computer was switched off - automatically requests all the missing transactions once the connection is available again until its local transaction ID corresponds to that of the Primary Directory Node.

3.4.2 Analysis of listener/notifier problems

Log files/debug level of replication

All status messages from the Univention Directory Listener and the executed listener modules are logged in the `/var/log/univention/listener.log` file. The level of detail of the log messages can be configured using the Univention Configuration Registry Variable *listener/debug/level* (page 297).

Status messages from the Univention Directory Notifier service are logged in the `/var/log/univention/notifier.log` file. The debug level can be configured using the *notifier/debug/level* (page 300).

Identification of replication problems

When the domain replication is running normally (normal system load, no network problems), the delay between changes being made in UMC modules and these changes being replicated to, for example, a Replica Directory Node is barely noticeable. An incomplete replication can be identified by comparing the transaction IDs of the listener and notifier services.

The transactions registered by the notifier service are written in the `/var/lib/univention-ldap/notify/transaction` file in ascending order on the Primary Directory Node. An example:

```
root@primary:~# tail -1 /var/lib/univention-ldap/notify/transaction
836 cn=replica3,cn=dc,cn=computers,dc=firma,dc=de m
```

The last transaction received by the listener system is stored in the `/var/lib/univention-directory-listener/notifier_id` file:

```
root@replica1:~# cat /var/lib/univention-directory-listener/notifier_id
836
```

This check can also be performed automatically by the Nagios service `UNIVENTION_REPLICATION` (see *Preconfigured Nagios checks* (page 288)).

Reinitialization of listener modules

If there are problems in running a listener module, there is the option to reinitialize the module. In this case, all LDAP objects with which the listener module works are passed on again.

Warning: This is a destructive operation. It removes some internal state of the listener. Use with care!

The name of the listener module must be supplied to the command for the renewed initialization. The installed listener modules can be found in the `/var/lib/univention-directory-listener/handlers/` directory.

The following command can be used to reinitialize the printer module, for example:

```
$ univention-directory-listener-ctrl resync cups-printers
```

3.5 SSL certificate management

In UCS, sensitive data are always sent across the network encrypted, e.g., via the use of SSH for the login to systems or via the use of protocols based on SSL/TLS. (*Transport Layer Security (TLS)* is the current protocol name, the name of the previous protocol *Secure Socket Layer (SSL)*, however, is still more common and is also used in this documentation).

For example, SSL/TLS is employed in the listener/notifier domain replication or for HTTPS access to UCS web interfaces.

Both communication partners must be able to verify the authenticity of the key used for encrypted communication between two computers. To this end, each computer also features a so-called *host certificate*, which is issued and signed by a certification authority (CA).

UCS provides its own CA, which is automatically set up during installation of the Primary Directory Node and from which every UCS system automatically procures a certificate for itself and the CA's public certificate when joining the domain. This CA appears as the root CA, signs its own certificate and can sign certificates for other certification authorities.

The properties of the CA are generated automatically during the installation based on system settings such as the locale. These settings can be subsequently adapted on the Primary Directory Node in the UMC module *Certificate settings*.

Caution: If the UCS domain contains more than one system, all other host certificates need to be reissued after changing the root certificate! The procedure required for this is documented in KB 37 - Renewing the SSL certificates[9].

The UCS-CA is always found on the Primary Directory Node. A copy of the CA is stored on every Backup Directory Node, which is synchronized with the CA on the Primary Directory Node by a Cron job every 20 minutes.

Caution: The CA is synchronized from the Primary Directory Node to the Backup Directory Node and not vice-versa. For this reason, only the CA on the Primary Directory Node should be used.

If a Backup Directory Node is promoted to the Primary Directory Node (see *Converting a Backup Directory Node backup to the new Primary Directory Node* (page 49)), the CA on the new Primary Directory Node can be used directly.

The UCS root certificate has a specified validity period - as do the computer certificates created with it.

Caution: Once this period of time elapses, services which encrypt their communication with SSL (e.g., LDAP or domain replication) no longer function.

It is thus necessary to verify the validity of the certificate regularly and to renew the root certificate in time. A Nagios plugin is provided for the monitoring of the validity period. In addition, a warning is shown when opening a UMC module if the root certificate is going to expire soon (the warning period can be specified with the Univention Configuration Registry Variable `ssl/validity/warning` (page 305); the standard value is 30 days).

The renewal of the root certificate and the other host certificates is documented in KB 37 - Renewing the SSL certificates[10].

On UCS systems, a Cron job verifies the validity of the local computer certificate and the root certificate daily and records the expiry date in the Univention Configuration Registry variables `ssl/validity/host` (page 305) (host certificate) and `ssl/validity/root` (page 305) (root certificate). The values entered there reflect the number of days since the 1970-01-01.

[9] https://help.univention.com/t/37
[10] https://help.univention.com/t/37

In Univention Management Console, the effective expiry date of the computer and root certificate can be accessed via the upper right menu and the entry *License ▸ License information*.

3.6 Kerberos

Kerberos is an authentication framework the purpose of which is to permit secure identification in the potentially insecure connections of decentralized networks. In Kerberos, all clients use a foundation of mutual trust, the *Key Distribution Center* (KDC). A client authenticates at this KDC and receives an authentication token, the so-called ticket which can be used for authentication within the Kerberos environment (the so-called Kerberos realm). The name of the Kerberos realm is configured as part of the installation of the Primary Directory Node and stored in the Univention Configuration Registry Variable *kerberos/realm* (page 295). It is not possible to change the name of the Kerberos realm at a later point in time.

Tickets have a standard validity period of 8 hours; this is why it is vital for a Kerberos domain to have the system time synchronized for all the systems belonging to the Kerberos realm.

Univention Corporate Server uses the Heimdal Kerberos implementation. An independent Heimdal service is started on UCS Directory Nodes without Samba/AD, while Kerberos is provided by a Heimdal version integrated in Samba on Samba/AD DCs. In a environment composed of UCS Directory Nodes without Samba/AD and Samba/AD domain controllers both Kerberos environments are based on identical data (these are synchronized between Samba/AD and OpenLDAP via the Univention S4 connector (see *Univention S4 connector* (page 173))).

3.6.1 KDC selection

As standard, the KDC is selected via a DNS service record. The KDC used by a system can be reconfigured using the Univention Configuration Registry Variable *kerberos/kdc* (page 295). If Samba/AD is installed on a system in the domain, the service record is reconfigured so that only the Samba/AD-based KDCs are offered. In a mixed environment it is recommended only to use the Samba/AD KDCs.

3.6.2 Kerberos admin server

The Kerberos admin server, on which the administrative settings of the domain can be made, runs on the Primary Directory Node. Most of the settings in Univention Corporate Server are taken from the LDAP directory, so that the major remaining function is changing passwords. This can be achieved by means of the Tool **kpasswd**; the passwords are then changed in the LDAP too. The Kerberos admin server can be configured on a system via the Univention Configuration Registry Variable *kerberos/adminserver* (page 295).

3.7 Password hashes in the directory service

User password hashes are stored in the directory service in the `userPassword` attribute. The **crypt** library function is used to hash passwords. The actual hashing method can be configured via the Univention Configuration Registry Variable *password/hashing/method* (page 302), `SHA-512` is used by default.

As an alternative Univention Corporate Server (from version UCS 4.4 erratum 887[11] on) offers the option of using **bcrypt** as hashing method for passwords of user accounts. To activate **bcrypt** support in OpenLDAP the Univention Configuration Registry Variable *ldap/pw-bcrypt* (page 297) has to bet set to `true` on all LDAP servers. Otherwise it is not possible authenticate with a **bcrypt** hash as password hash. Additionally the Univention Configuration Registry Variable *password/hashing/bcrypt* (page 301) has to be set to `true`, again on all servers, to activate **bcrypt** as the hashing method for setting or changing user password.

In addition, the **bcrypt** cost factor and the **bcrypt** variant can be configured via the Univention Configuration Registry Variables *password/hashing/bcrypt/cost_factor* (page 302) (default `12`) and *password/hashing/bcrypt/prefix* (page 302) (default `2b`).

[11] https://errata.software-univention.de/#/?erratum=4.4x887

Caution: **bcrypt** is limited to a maximum of 72 characters. So only the first 72 characters of the password are used to generate the hashes.

3.8 SAML identity provider

SAML (Security Assertion Markup Language) is an XML-based standard for exchanging authentication information in order to allow single sign-on across domain boundaries. UCS provides a fail-safe SAML identity provider on a Primary Directory Node as well as Backup Directory Node. The SAML identity provider is registered at an external service with a cryptographic certificate and establishes a trust relationship. The user then only needs to authenticate himself against UCS and can use the service without renewed authentication.

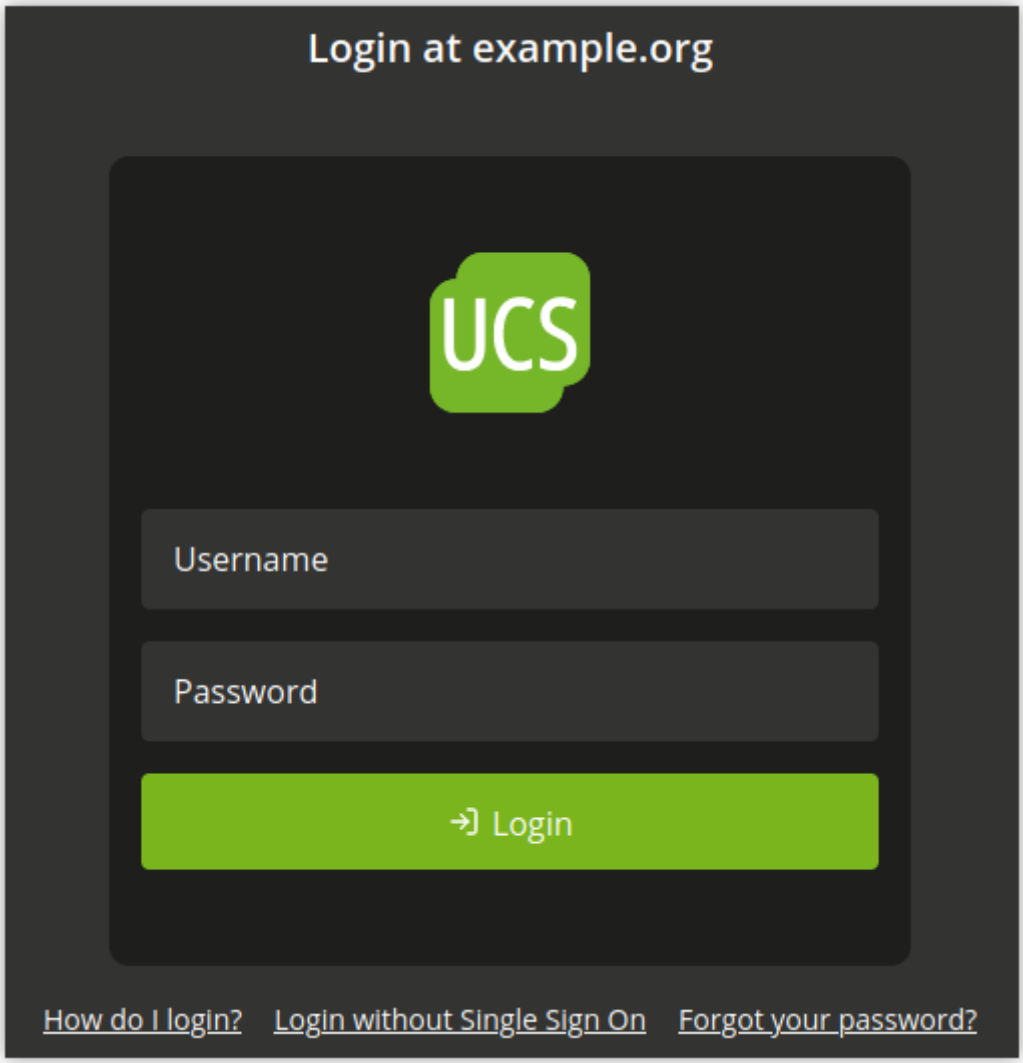


Fig. 3.3: The single sign-on login page

The SAML 2.0 compatible UCS identity provider is provided by the integration of **simplesamlphp**.

The UCS identity provider is tightly integrated into the UCS domain. Clients that will be used to access the UCS identity provider have to be able to resolve DNS records in the UCS domain. The domain DNS Servers should therefore be configured on all clients in order to be able to resolve the central DNS record, which by default is `ucs-sso.[Domain name]`.

The UCS identity provider is automatically installed on Primary Directory Node and Backup Directory Nodes. Further Backup Directory Nodes can be made available in the domain to increase fail-safe safety. The default DNS record `ucs-sso.[Domain name]` is registered to increase fail-safe access to the UCS identity provider. The SSL certificate for this record is kept on all participating systems in the domain. It is advised to install the UCS domain root certificate on all clients that are using *single sign-on*.

It is possible to associate the SAML authentication with the Kerberos login. This means that users with a valid Kerberos ticket, for example after logging on to Windows or Linux, can sign in to the identity provider without having to manual re-authenticate.

To allow Kerberos authentication at the identity provider, the Univention Configuration Registry Variable *saml/idp/authsource* (page 304) has to be changed from `univention-ldap` to `univention-negotiate`. The web browsers must be configured to transfer the Kerberos ticket to the SAML Identity Provider. Here are two examples for the configuration of Firefox and Internet Explorer / Microsoft Edge:

Mozilla Firefox
: In the extended Firefox configuration, which can be reached by entering `about:config` in the Firefox address line, the address of the identity provider must be entered in the option `network.negotiate-auth.trusted-uris`, which is `ucs-sso.[Domain name]` by default.

Microsoft Internet Explorer; Microsoft Edge
: In the Control Panel, the *Internet Options* must be opened, followed by *Security* ‣ *Local Intranet* ‣ *Sites* ‣ *Advanced*. The address of the identity provider has to be added, which is `ucs-sso.[Domain name]` by default.

The Kerberos authentication can be restricted to certain IP subnets by setting the Univention Configuration Registry Variable *saml/idp/negotiate/filter-subnets* (page 304) for example to `127.0.0.0/16,192.168.0.0/16`. This is especially useful to prevent a pop up login box being shown for clients which are not part of the UCS domain.

3.8.1 Login via single sign-on

The activation of *single sign-on* for the portal is described in *Login* (page 57). For this, `ucs-sso.[Domain name]` must be reachable. To login the domain credentials must be provided. For the login directly at the UCS system (i.e., without *single sign-on*), follow the link *Login without Single Sign On*.

The design of the login dialog can be changed by editing `/usr/share/univention-management-console-login/css/custom.css`. This file will never be altered or deleted during updates.

Other web services will redirect to the UCS identity provider login page in a similar fashion in order to carry out a *single sign-on*. After authenticating, the user will be forwarded back to the web service itself. These services need to be registered as described in *Adding a new external service provider* (page 46).

The *single sign-on* for a particular service can be initiated from the UCS identity provider, as well. This saves an extra visit at the external web service which redirects to the authentication site. To do so, a link to the UCS identity provider page needs to be provided in the form of `https://ucs-sso.[Domain name]/simplesamlphp/saml2/idp/SSOService.php?spentityid=[Service provider identifier]`.

3.8.2 Adding a new external service provider

The UMC module *SAML identity provider* allows to manage all service providers that are registered at the UCS identity provider. Users have to be activated for a service provider, to be able to authenticate for it at the UCS identity provider. The service provider can be activated for groups, to allow authentication with that service provider for all users within that group. On the user's *Account* tab or the group's *General* tab, the service provider can to be added under *SAML settings*.

To register the UCS identity provider at an external service provider, the public part of the SAML certificate is required by the service provider. The certificate can be downloaded via a link in the UMC module. Some service providers may require the UCS identity provider XML metadata as a file upload. By default the XML file can

be downloaded from the URL `https://ucs-sso.[Domain name]/simplesamlphp/saml2/idp/metadata.php`.

The following attributes can be configured when adding a new service provider.

Table 3.1: General options when configuring a service provider

Attribute	Description
Service provider activation status	If activated, the configuration for the service provider is activated and is ready for authentication.
Service provider identifier	Defines the internal name of the service provider. The name is later selected at user objects, when giving them access to a service provider. The identifier cannot be changed later.
Respond to this service provider URL after login	After successful authentication, the user's browser is redirected to the service provider. The redirection is done to this provided URL.
Single logout URL for service provider	Service providers can offer a URL endpoint at which the session at the service provider can be terminated. If a user logs out at the UCS identity provider, the browser will get redirected to the provided URL to terminate the session.
Format of `NameID` attribute	The value `NameIDFormat` that the service provider receives. The service provider's documentation should contain information about possible values. Example: `urn:oasis:names:tc:SAML:2.0:nameid-format:transient` or `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`.
Name of the attribute that is used as `NameID`	The LDAP attribute that is used to uniquely identify the user is provided here, e.g., `uid`.
Name of the organization for service provider	The value provided here will be shown on the UCS single sign-on login page. It helps the user to identify for which service they enter credentials.
Description of this service provider	The value provided here will be shown on the UCS single sign-on login page. A longer description about the service provider can be given here. The description will be shown on the login page in a separate paragraph.

Table 3.2: Advanced settings when configuring a service provider

Attribute	Description
URL to the service provider's privacy policy	If a URL is entered here, the UCS identity provider login page will contain a link to this URL.
Allow transmission of LDAP attributes to the service provider	By default, the UCS identity provider transmits only the `NameID` attribute entered on the *General* page to the service provider. If additional LDAP user attributes are required by the service provider, this checkbox can be activated. The attributes that should be transmitted have to be entered in the *List of LDAP attributes to transmit*.
Value for `attribute format` field	In case the transmitted attributes need to be sent in a particular format value, this format can be entered here. Example: `urn:oasis:names :tc:SAML:2.0:nameid-format:transient` or `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`.
List of LDAP attributes to transmit	Every LDAP attribute that should be transmitted to the service provider can be entered here. Additionally, one or more service attribute names can be added to each LDAP attribute in the field next to it. These service attribute names have the purpose to translate the LDAP Attribute names for the service provider. Multiple service attribute names have to be separated by commas. In order for the UCS identity provider to process these attributes, they need to be registered additionally via the LDAP object `id=default-saml-idp,cn=univention,[LDAP base DN]`. LDAP attributes entered at the object can be read and transferred by the Identity Provider.

3.8.3 Extended Configuration

Some environments may require the UCS Identity Provider to provide multiple logical Identity Provider instances. Logical separation is achieved by offering different URIs as Identity Provider endpoints.

The default endpoint is `https://ucs-sso.[Domain name]/simplesamlphp/saml2/idp/metadata.php`. Further entries can be created by setting Univention Configuration Registry Variables in the form *saml/idp/entityID/supplement/[identifier]* (page 304) to `true` on all servers which serve the UCS Identity Provider. Typically that will be the Primary Directory Node and all Backup Directory Nodes. The **apache2** service must then be reloaded.

For example, to set up another entry under the URI `https://ucs-sso.[Domain name]/simplesamlphp/[secondIDP]/saml2/idp/metadata.php`, the Univention Configuration Registry Variable `saml/idp/entityID/supplement/secondIDP=true` must be set.

3.9 OpenID Connect Provider

UCS offers the possibility to install a *OpenID Connect Provider*, which allows external web services to delegate the user login via the *OpenID Connect (OIDC)* protocol to the UCS Identity Management. The **OpenID Connect Provider** App can be installed via the App Center. The service is provided by the software **Kopano Konnect**.

The app can be installed on all system roles. When installing on a UCS system with the role Primary Directory Node or Backup Directory Node the **OpenID Connect Provider** is made available under the DNS entry for the single sign-on, normally this is `ucs-sso.Domain name`.

If the app is installed on a different system role, the provider can be reached directly via the hostname instead. It should be ensured that the app is installed on all other servers that are reachable at the `ucs-sso` DNS CNAME.

Session synchronization between multiple installed OIDC Providers in a domain is not preconfigured. When experiencing login issues with Apps, we recommend to only install the OIDC Provider on one system, and restrict the `ucs-sso` DNS CNAME to that system, or contact Univention Support.

External Web services can be connected to UCS via **OpenID Connect** by creating a specific object of type `oidc/rpservice` for this service in the UCS directory service. These can be created via the UMC module *LDAP directory* in the container `cn=oidc`, which is located below the container `cn=univention`. Here the new service can be registered via the item Add and the selection *OpenID Connect Relying Party Service*.

The same is also possible from the command line:

```
$ udm oidc/rpservice create --set name=$UCS_internal_identifier \
  --position="cn=oidc,cn=univention,$(ucr get ldap/base)" \
  --set clientid="$ClientID" \
  --set clientsecret="$a_long_password" \
  --set trusted=yes \
  --set applicationtype=web \
  --set redirectURI="$URL_from_services_documentation"
```

The command parameters are:

name
: the service name displayed in the web interface during login.

clientid
: must be identical here and in the connected service (*shared secret*).

secret
: must be identical here and in the connected service (*shared secret*).

trusted
: should be set to `yes` by default. Otherwise, the user will be prompted for confirmation to transfer their user attributes to the service.

applicationtype
should be set to `web` for internet services.

redirectURI
URL of the login endpoint, which can be found in the documentation of the connected service. If a service is accessible via several URLs or should it also be accessible via IP address, all possible addresses must be added to the `redirectURI` attribute. The field can therefore be defined multiple times, whereby each individual value must contain a valid URL.

The connected web service still needs information about the *OpenID Connect* endpoints of the provider app for its configuration. If the provider app is installed, this information can be found at the URL `https://ucs-sso[Domain name]/.well-known/openid-configuration`. If the provider app was installed on a system other than Primary Directory Node or Backup Directory Node, use the FQDN of the respective server instead of `ucs-sso.Domain name` as described above.

When using *OpenID Connect*, resolvable DNS names and verifiable certificates are a prerequisite. This is especially true for client computers of end users who need to access both the DNS resolvable host names of the Web service and the OpenID Connect Provider. In addition, the externally connected Web services must be able to establish a connection to the OpenID Connect Provider in order to be able to retrieve the user attributes.

In the special case where the DNS name of the OIDC provider is to be changed, the corresponding value must first be adjusted in the app settings of the **OpenID Connect Provider** app. Since there are diverse scenarios for the availability of the provider after changing the DNS name, the web server configuration cannot be changed automatically. For example, depending on the configured DNS name, the UCS Apache configuration has to be adapted. The configuration file `/etc/apache2/conf-available/openid-connect-provider.conf` must be made available under the set DNS name in a virtual host.

With version 2 of the OIDC-Provider App the authentication to OpenID Connect works via the SAML Identity Provider of the UCS domain. If the SAML Identity Provider is not reachable at the default URL `https://ucs-sso.[Domain name]`, the correct URL under which the SAML IdP metadata for the UCS domain can be retrieved must be entered correctly in the app settings. If this URL is configured incorrectly, the OpenID Connect Provider will not start.

With SAML authentication, the authorization for the use of the OpenID Connect Provider and thus for all apps connected via OIDC can be controlled via SAML authorizations. By default, the group `Domain Users` is enabled for access when the app is installed. If this permission should be removed, the corresponding option must also be activated in the app settings so that the permission is not automatically added again.

The OpenID Connect Provider logs actions via the Docker Daemon. The output can be viewed with the command **univention-app logs openid-connect-provider**.

3.10 Converting a Backup Directory Node backup to the new Primary Directory Node

A UCS domain consists of only one Primary Directory Node, but is not limited in the number of Backup Directory Node. A Backup Directory Node stores all the domain data and all SSL security certificates as read-only copies. However, in contrast to the Primary Directory Node, writing changes are not allowed.

Any Backup Directory Node can be converted to a Primary Directory Node. There are two typical scenarios for this:

- In an emergency if the hardware of the Primary Directory Node fails.
- To replace a fully functional Primary Directory Node with new hardware or changing the architecture from *i386* to *amd64*.

Caution: The conversion of a Backup Directory Node to a Primary Directory Node is a serious configuration change and should be prepared carefully. The conversion cannot be reversed.

The Primary Directory Node that is going to be replaced has to be shut down before the conversion. It must not be powered on during or after the conversion!

Before the conversion, the installed software packages and the current configuration has to be compared between the Primary Directory Node and Backup Directory Node. If the Primary Directory Node is not available anymore, use a file backup. After the conversion, all possibly remaining references of the old Primary Directory Node have to be removed or changed to the new Primary Directory Node.

The conversion primarily involves the changeover of the services relevant for authentication such as LDAP, DNS, Kerberos and Samba. The installed software needs to be adjusted manually (this can be done using the UMC modules *App Center* or *Package Management*).

For example, if the mail component was installed on the previous Primary Directory Node, it will not be automatically installed on the new Primary Directory Node after the conversion. To minimize manual changes after the conversion, please consider *Fault-tolerant domain setup* (page 51).

If additional LDAP schema packages were installed on the Primary Directory Node, they must also be installed on the Backup Directory Node prior to the conversion. The package list of the old Primary Directory Node should be saved prior to the promotion in order to allow a subsequent comparison of the installed packages. The package list can be created with the following command:

```
$ dpkg --get-selections \* > dpkg.selection
```

Compare this file with the same output on the Backup Directory Node. The package list in the files should only differ in the packages **univention-server-master** and **univention-server-backup**. You must then install missing packages on the Backup Directory Node. Especially those packages that install a LDAP schema are absolutely necessary. The following command executed on the Primary Directory Node lists all packages with a LDAP schema:

```
$ dpkg -S /etc/ldap/schema/*.schema \
  /usr/share/univention-ldap/schema/*.schema
```

To simply install all packages of the Primary Directory Node also on the Backup Directory Node, use the previously created file `dpkg.selection` of the Primary Directory Node and run the following command on the Backup Directory Node:

```
$ dpkg --set-selections < dpkg.selection
$ apt-get dselect-upgrade
```

In addition, the Univention Configuration Registry inventory needs to be saved so that it is possible to compare the configuration adjustments on the new Primary Directory Node. The following files on the Primary Directory Node need to be compared with those on the Backup Directory Node:

- `/etc/univention/base.conf`
- `/etc/univention/base-forced.conf`

UCS saves a copy of those files every night to `/var/univention-backup/ucr-backup_%Y%m%d.tgz`.

The conversion of a Backup Directory Node to the new Primary Directory Node is performed by running the command **/usr/lib/univention-ldap/univention-backup2master** on the Backup Directory Node. The system must be rebooted after the conversion. The process is logged to `/var/log/univention/backup2master.log` The following steps are performed by **univention-backup2master**:

1. Checking the environment: The system must be a Backup Directory Node that already joined the domain. Additionally, it is checked if the Primary Directory Node can be resolved via DNS and if the repository server can be reached. Also, the Primary Directory Node must be powered off and not reachable anymore.
2. Now, the most important services OpenLDAP, Samba, Kerberos and Univention Directory Notifier and Listener will be stopped. Important Univention Configuration Registry Variable, such as *ldap/master* (page 296) and *server/role* (page 304) will be changed. The UCS Root CA certificate will be available via the web server on the Backup Directory Node. All mentioned services will be started again.
3. The DNS SRV record `kerberos-adm` will be changed from the old to the new Primary Directory Node.

4. If present, the Univention S4 Connector (see *Univention S4 connector* (page 173)) will be removed from the computer object of the old Primary Directory Node and will be scheduled for re-configuration on the new Primary Directory Node.
5. The server role of the new Primary Directory Node will be changed to `domaincontroller_master` in the OpenLDAP directory service. The DNS SRV record `_domaincontroller_master._tcp` will also be adjusted.
6. If present, all entries of the old Primary Directory Node will be removed from the local Samba directory service. Additionally, the FSMO roles will be transferred to the new Primary Directory Node.
7. The computer object of the old Primary Directory Node will be deleted from OpenLDAP.
8. The OpenLDAP directory service will be searched for any remaining references to the old Primary Directory Node. All found references (e.g. DNS records) are shown and suggested to be fixed. The suggested fixes have to be checked and confirmed one by one.
9. Finally, the package **univention-server-backup** will be replaced by **univention-server-master**.

Subsequently, the LDAP directory on the new Primary Directory Node and the Univention Configuration Registry values on all UCS systems of the domain should be checked for any remaining references to the hostname or the IP address of the old Primary Directory Node. Those references need to be adjusted to the new Primary Directory Node, too.

For additional details, see Univention Help 19514 - "How To: backup2master"[12].

3.11 Fault-tolerant domain setup

In a domain exist some services that are important for the functionality of all of its members. Redundancy can be used to remove those single points of failure. An article in the Univention Support database explains how to secure LDAP, Kerberos, DNS, DHCP and Active Directory-compatible Domain Controllers: KB 6682 - Fail-safe domain setup[13].

3.12 Protocol of activities in the domain

The **Admin Diary** app provides the facility to log important events happening in the domain. This includes among others:

- Creation, move, modification and deletion of users and other objects using Univention Directory Manager
- Installation, update and deinstallation of apps
- Server password changes
- Start, end and eventual failures of domain joins
- Start and end of UCS updates

Fig. 3.4 shows, how events are shown in the UMC module *Admin Diary*. By default the displayed entries are grouped by week and can additionally be filtered through the search field. Selecting an entry from the list opens a dialog showing additional details about the who and when of the event, as shown in Fig. 3.5. Moreover there is the possibility to comment each event.

The app consists of two components:

Admin Diary backend
: The backend must be installed on one system in the domain before the frontend can be installed. It includes a customization for **rsyslog** and writes into a central database, which defaults to PostgreSQL. If MariaDB or MySQL is already installed on the target system, it will be used instead of PostgreSQL.

[12] https://help.univention.com/t/19514
[13] https://help.univention.com/t/6682

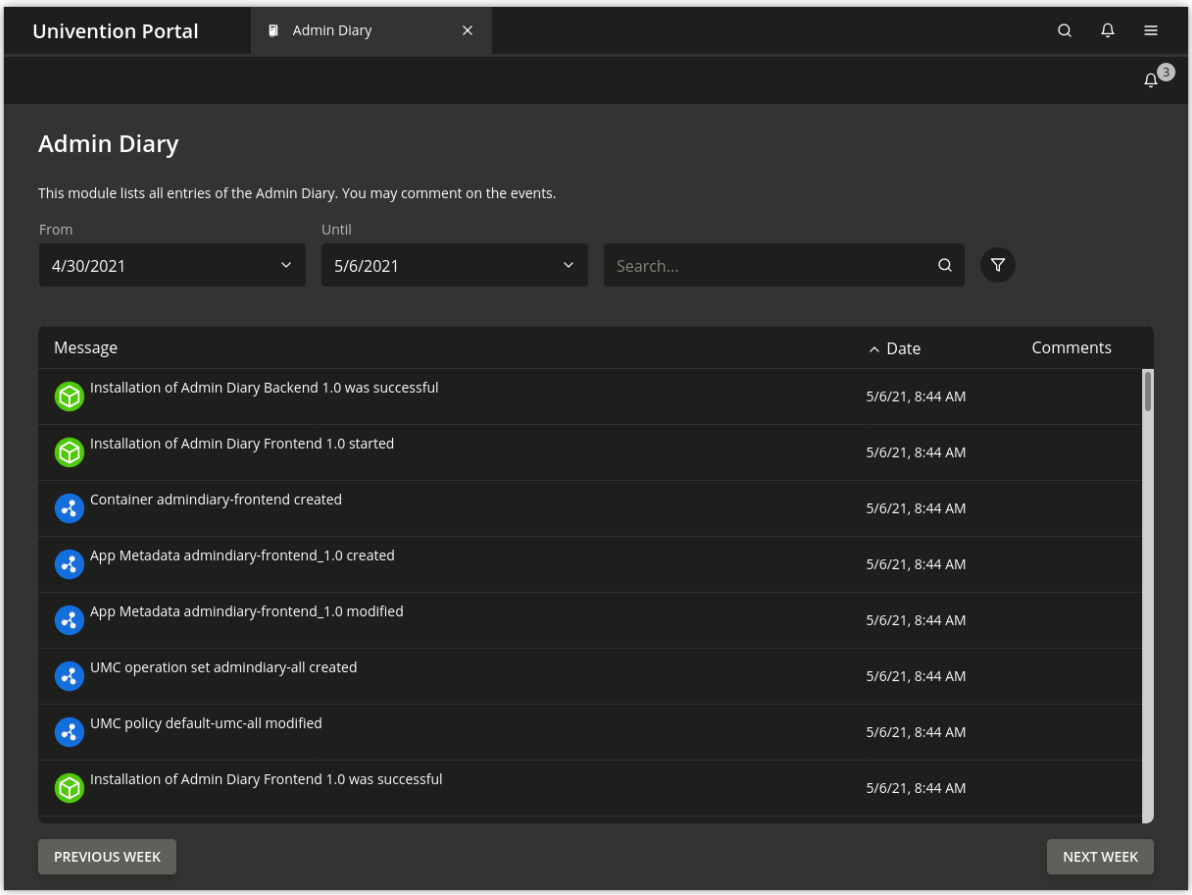


Fig. 3.4: View of events in Admin Diary

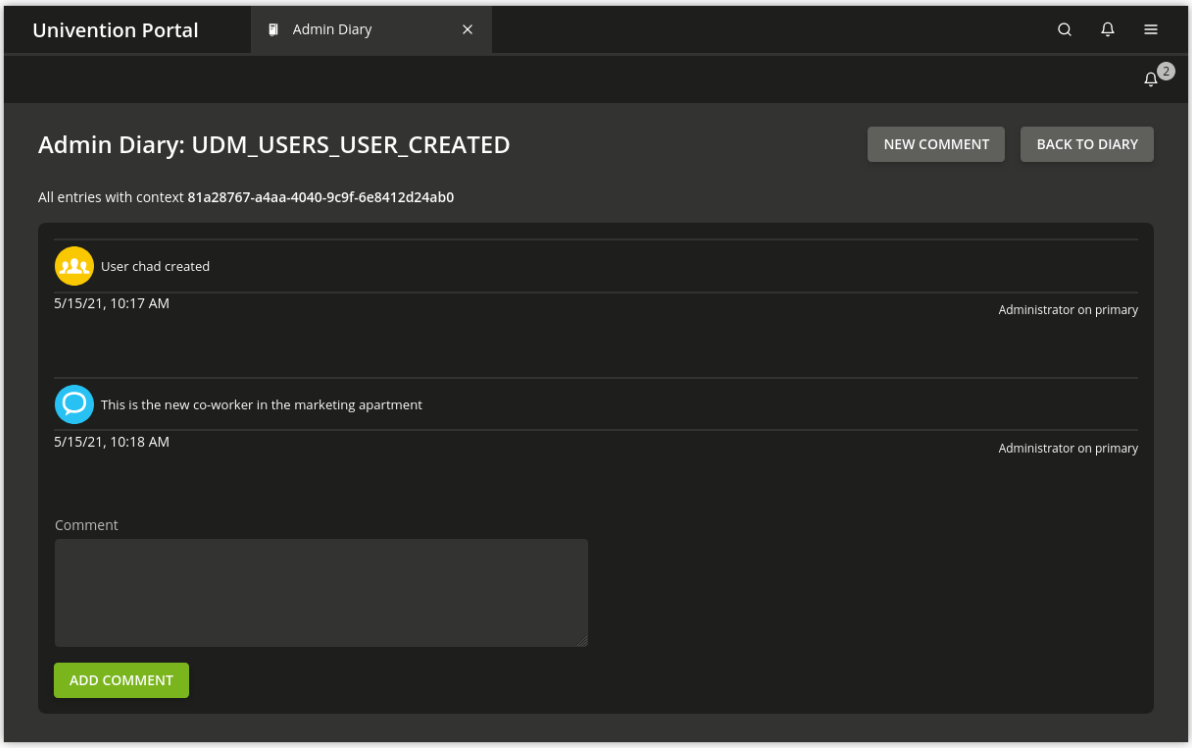


Fig. 3.5: Detail view in Admin Diary

Admin Diary frontend

Likewise the frontend must be installed at least once, but more installations are also possible. The frontend includes the UMC module *Admin Diary*, which is used to show and comment the entries. When installing it on a different host than where the backend is installed, access to the central database needs to be configured manually. The required steps for this are described in Admin Diary - How to separate frontend and backend[14].

[14] https://help.univention.com/t/admin-diary-how-to-seperate-frontend-and-backend/11331

CHAPTER

FOUR

UCS WEB INTERFACE

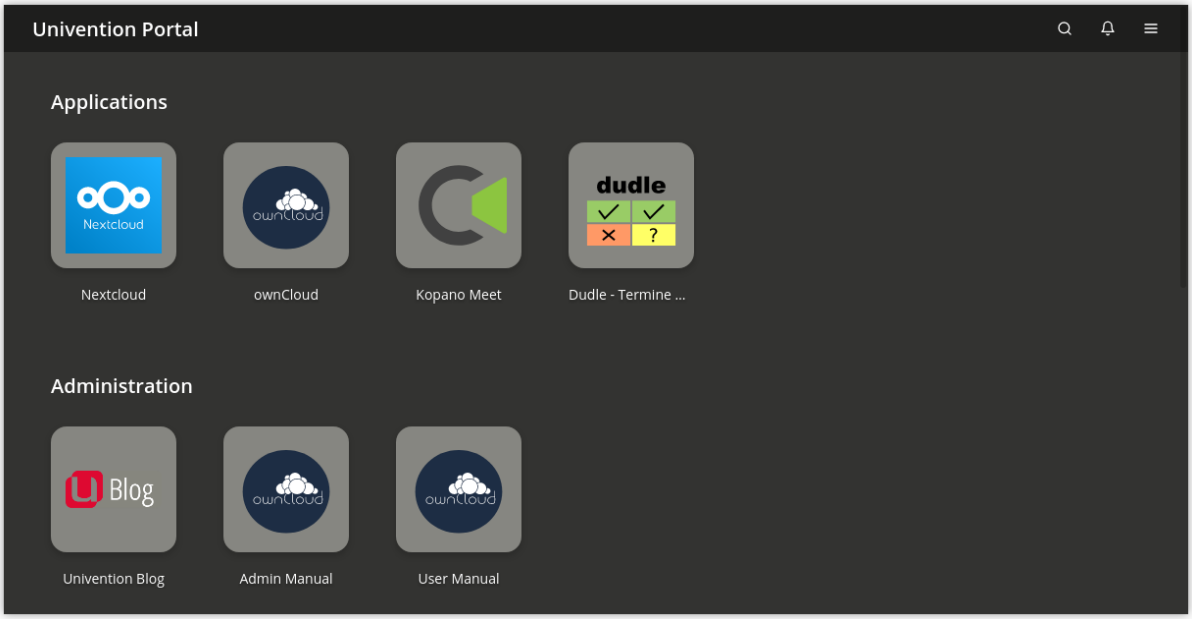


Fig. 4.1: UCS portal page

The UCS web interface is the central tool for managing a UCS domain as well as for accessing installed applications of the domain.

The UCS web interface is divided into several pages which all have a similarly designed header. Via the symbols in the top right, one may launch a search on the current page (magnifier) or open the user menu (three bars) (login is possible through the latter). The login at the web interface is done via a central page once for all sub pages of UCS as well as for third party applications as far as a web based *single sign-on* is supported (*Login* (page 57)).

Central starting point for users and administrators for all following actions is the UCS portal page (cf. Fig. 4.1). By default, the portal page is available on all system roles and allows an overview of all Apps and further services which are installed in the UCS domain. All aspects of the portal page can be customized to match one's needs (*UCS portal page* (page 65)).

For environments with more than one server, an additional entry to a server overview page is shown on the portal page. This sub page gives an overview of all available UCS systems in the domain. It allows a fast navigation to other systems in order to adjust local settings via UMC modules.

UMC modules are the web based tool for the administration of the UCS domain. There are various modules available for the administration of the different aspects of a domain depending on the respective system role. Installing additional software components may add new UMC modules to the system. *Univention Management Console modules* (page 69) describes their general operation.

The subsequent sections detail the usage of various aspects of the domain management. *LDAP directory browser* (page 73) gives an overview of the LDAP directory browser. The use of administrative settings via policies is discussed in *Policies* (page 74). How to extend the scope of function of the domain administration is detailed in *Expan-*

sion of UMC modules with extended attributes (page 76). *Structuring of the domain with user-defined LDAP structures* (page 80) details how containers and organizational units can be used to structure the LDAP directory. *Delegated administration for UMC modules* (page 81) explains delegating administration rights to additional user groups.

In conclusion, the command line interface of the domain administration is illustrated (*Command line interface of domain management (Univention Directory Manager)* (page 81)), and the evaluation of domain data via the UCS reporting function are explained (*Evaluation of data from the LDAP directory with Univention Directory Reports* (page 88)).

Note: The UCS web interface is part of Univention Nubus in the *Management UI* component. For more information about Nubus, refer to *What is Univention Nubus?* (page 2)

4.1 Introduction

4.1.1 Access

The UCS web interface can be opened on any UCS system via the URL `https://servername/`. Alternatively, access is also possible via the server's IP address. Under certain circumstances it may be necessary to access the services over an insecure connection (e.g., if no SSL certificates have been created for the system yet). In this case, `http` must be used instead of `https` in the URL. In this case, passwords are sent over the network in plain text!

4.1.2 Browser compatibility

The UCS web interface uses numerous JavaScript and CSS functions. Cookies need to be permitted in the browser. The following browsers are supported:

- **Chrome** as of version 85
- **Firefox** as of version 78
- **Microsoft Edge** as of version 88
- **Safari** and **Safari Mobile** as of version 13

Users with older browsers may experience display problems or the site does not work at all.

The UCS web interface is available in German and English (and French if it is chosen as language during the installation from DVD); the language to be used can be changed via the entry *Switch language* of the user menu in the upper right corner.

4.1.3 Switching between dark and light theme for UCS web interfaces

All UCS web interfaces have a dark and a light theme that can be switched between with the Univention Configuration Registry Variable *ucs/web/theme* (page 306). The value of *ucs/web/theme* (page 306) corresponds to a CSS file under `/usr/share/univention-web/themes/` with the same name (without file extension). For example, setting *ucs/web/theme* (page 306) to `light` will use `/usr/share/univention-web/themes/light.css` as theme for all UCS web interfaces.

4.1.4 Creating a custom theme/Adjusting the design of UCS web interfaces

To customize a theme for UCS web interfaces don't edit the files `/usr/share/univention-web/themes/dark.css` and `/usr/share/univention-web/themes/light.css`, because UCS upgrades can overwrite your changes. Instead, copy one of these files to, for example, `/usr/share/univention-web/themes/mytheme.css` and set the Univention Configuration Registry Variable *ucs/web/theme* (page 306) to `mytheme`.

The files `/usr/share/univention-web/themes/dark.css` and `/usr/share/univention-web/themes/light.css` contain the same list of CSS variables[15]. Other CSS files use these CSS variables. These CSS variables are the supported layer of configurability for UCS web interfaces. The names and use cases for these variables don't change between UCS upgrades, but Univention may add additional names and use cases.

Some UCS web interfaces import their own local `custom.css` file which you can use to adjust the design of the following pages:

- For *Login via single sign-on* (page 46): `/usr/share/univention-management-console-login/css/custom.css`
- For *UCS portal page* (page 65): `/usr/share/univention-portal/css/custom.css`

The files are empty during the installation of UCS. UCS updates don't change these files.

Important: Be aware, however, that a given CSS selector[16] may break when installing a UCS update.

4.1.5 Feedback on UCS

By choosing the *Help* ‣ *Feedback* option in the upper right menu, you can provide feedback on UCS via a web form.

4.1.6 Collection of usage statistics

Anonymous usage statistics on the use of the UCS web interface are collected when using the *core edition* version of UCS (which is generally used for evaluating UCS). Further information can be found in KB 6701 - Data collection in Univention Corporate Server[17].

4.2 Login

UCS provides a central login page. You can sign in to the UCS web interface with the credentials of the respective domain account. On the portal at `https://FQDN/univention/portal/` you can use the following ways to sign in:

- Click the tile *Login* on the portal page.
- Go to *Menu* ‣ *Login*.

It opens the login page as shown in Fig. 4.2.

If a page in the management system, such as a UMC module, requires a login, your browser redirects you to the central login page.

When you sign in at the local UCS system, the browser session closes by default after 8 hours of inactivity. You can change the timeout through the UCR variable *umc/http/session/timeout* (page 306). To get a new the session, the user must sign in again.

[15] https://developer.mozilla.org/en-US/docs/Web/CSS/Using_CSS_custom_properties
[16] https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Selectors
[17] https://help.univention.com/t/6701

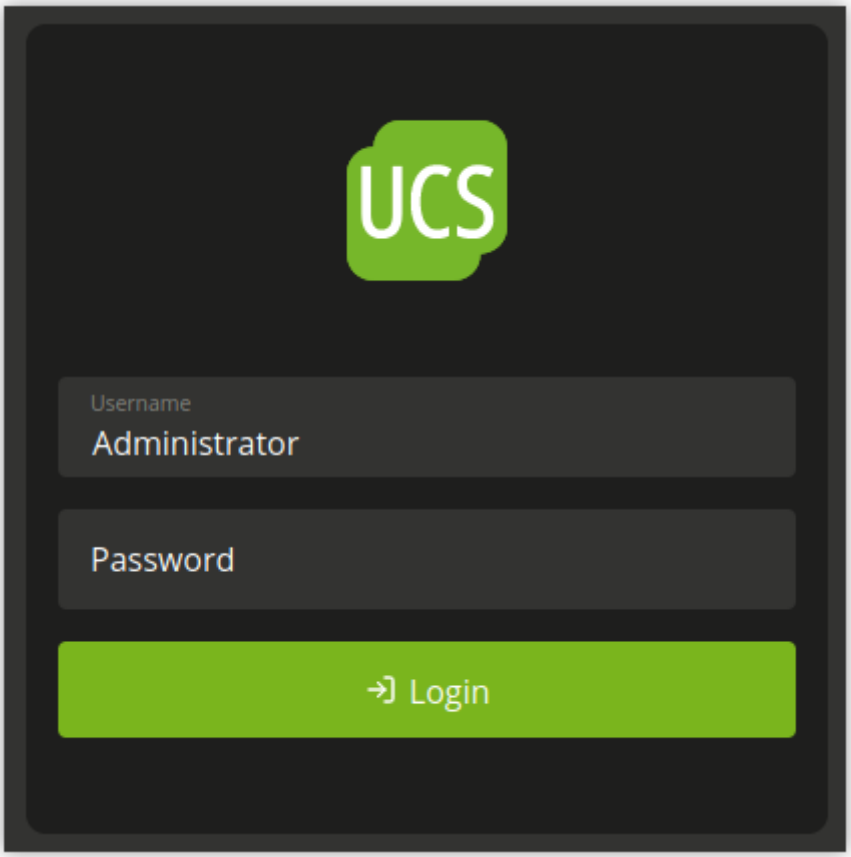


Fig. 4.2: UCS sign-in page

To sign out of the management system, click *Logout* in the user menu.

By installing a third-party application, such as **privacyIDEA**, it's possible to extend the UCS web interface authentication with a two-factor authentication (2FA).

4.2.1 Refresh portal tabs on logout

All browser tabs where the user is logged into the portal will automatically refresh when a logout is detected. This feature is deactivated by default and can be toggled through the UCR Variable `portal/reload-tabs-on-logout`.

4.2.2 Choose the right user account

To sign in, enter the *Username* and *Password* of the corresponding domain account in the login mask.

Administrator
: When you sign in with the `Administrator` account on a Primary Directory Node or Backup Directory Node, the UCS management system shows the UMC modules for the administration and configuration of the local system, as well as, UMC modules for the administration of data in the domain.

  You specified the initial password for the `Administrator` account in the setup wizard during installation. The password corresponds to the initial password of the local `root` account. Use the `Administrator` account for the initial sign in at a newly installed Primary Directory Node.

root
: In some cases, it might be necessary to sign in with the system's local `root` account. For more information, refer to *Administrative access with the root account* (page 162). The `root` account only enables access to UMC modules for the administration and configuration of the local system.

Other user accounts
: When you sign in with another user account, the UCS management system shows the UMC modules approved for the user. For additional information on allowing further modules, refer to *Delegated administration for UMC modules* (page 81).

4.2.3 Single sign-on

By default, the login page for the portal has single sign-on deactivated. The following sections describe how to activate single sign-on. After a successful sign in, the session is valid for all UCS systems of the domain, as well as, for third party apps, if the apps support web based single sign-on.

For sign-in through single sing-on, the browser session closes for 8 hours of inactivity. To get a fresh session, the user must sign in again.

It's possible to enforce the sign in on the local system by clicking the link *Login without Single Sign On* on the login page, as show in Fig. 4.3.

SAML for single sign-on

UCS has SAML activated by default. This section describes how to activate it for the *Login* buttons in the Portal. For more information about SAML, refer to *SAML identity provider* (page 45).

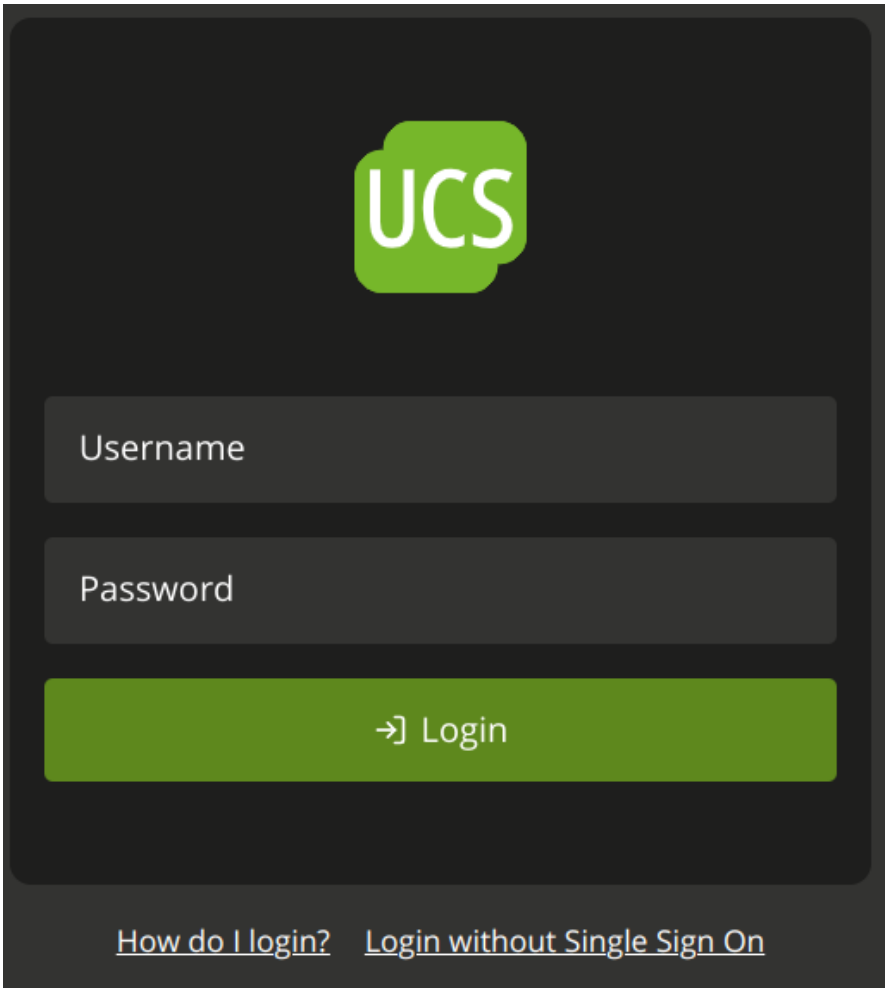


Fig. 4.3: UCS sign-in page for single sign-on

Activate

To activate single sign-on through SAML, use the following steps:

1. Ensure that all users in your domain who want to use the portal and the UCS management system with single sign-on can reach `ucs-sso.[Domain Name]`.
2. Change the Univention Configuration Registry Variable *portal/auth-mode* (page 302) to `saml` with *ucr set* (page 158). The default value was `ucs`.
3. For the change to take effect, restart the portal server with the following command:

```
$ systemctl restart univention-portal-server.service
```

Update sign-in links

Restarting the portal server automatically updates the *Login* link in the user menu. You must manually update the portal tile. The default portal has a preconfigured single sign-on login tile. Use the portal edit mode to enable it. To replace the *Login* tile with the single sign-on tile, follow these steps:

1. In *Univention Management Console* open the UMC Module Portal: *Domain* ‣ *Portal*.
2. To activate the preconfigured sign in tile for SAML, edit the entry `login-saml`, scroll down to the section *Advanced* and activate the checkbox *Advanced*.
3. To deactivate the default sign in tile, edit the entry `login-ucs`, scroll down to the section *Advanced* and deactivate the checkbox *Advanced*.

To change back to the default sign-in in UCS without single sign-on, you need to revert the steps for updating the portal tile and set the UCR variable *portal/auth-mode* (page 302) to `ucs`.

OpenID Connect for single sign-on

New in version 5.0-8-erratum-1118: With UCS 5.0 erratum 1118[18] the portal and the UCS management system have the capability to allow single sign-on with OpenID Connect. The capability is deactivated by default.

OpenID Connect (OIDC) is a protocol that allows single sign-on. OIDC is a more lightweight protocol than SAML. It's one variant for using single sign-on in the portal and the UCS management system. This section describes how to use it with UCS.

Requirements

Before you can use OIDC for single sign-on, you must meet the following requirements:

1. You must at least have UCS 5.0 erratum 1118[19] installed throughout your UCS domain.

   For information about how to upgrade, refer to *Updates of UCS systems* (page 96).

2. You must have the app **Keycloak** installed in your UCS domain.

   For information about the installation of **Keycloak**, refer to Installation[20] in *Univention Keycloak app documentation* [4].

[18] https://errata.software-univention.de/#/?erratum=5.0x1118
[19] https://errata.software-univention.de/#/?erratum=5.0x1118
[20] https://docs.software-univention.de/keycloak-app/latest/installation.html#app-installation

Activation

First, you need to decide on which UCS systems you want to enable single sign-on using OpenID Connect. Second, you need to apply the following steps to each of those UCS systems.

1. Deactivate SAML for portal sign-in through the UCR variable *umc/web/sso/enabled* (page 306) so that the automatic to sign in again doesn't try SAML first, but instead uses OIDC directly.

   Change the Univention Configuration Registry Variable *umc/web/oidc/enabled* (page 306) to `true` with *ucr set* (page 158).

   ```
   $ ucr set \
      umc/web/sso/enabled=false \
      umc/web/oidc/enabled=true
   ```

2. Run the join script for the UMC web server:

   ```
   $ univention-run-join-scripts \
      --force \
      --run-scripts \
      92univention-management-console-web-server.inst
   ```

3. Change the Univention Configuration Registry Variable *portal/auth-mode* (page 302) to `oidc` with *ucr set* (page 158). The default value was `ucs`.

4. For the change to take effect, restart the portal server with the following command:

   ```
   $ systemctl restart univention-portal-server.service
   ```

Create sign-in links

Restarting the portal server automatically updates the *Login* link in the user menu. You can optionally create a portal tile for the sign-in with OpenID Connect on the Primary Directory Node with the commands in Listing 4.1.

Listing 4.1: Create portal tile for sign-in with OpenID Connect

```
$ udm portals/entry create --ignore_exists \
    --position "cn=entry,cn=portals,cn=univention,$(ucr get ldap/base)" \
    --set name=login-oidc \
    --append displayName="\"en_US\" \"Login (Single sign-on)\"" \
    --append displayName="\"de_DE\" \"Anmelden (Single Sign-on)\"" \
    --append description="\"en_US\" \"Log in to the portal\"" \
    --append description="\"de_DE\" \"Am Portal anmelden\"" \
    --append link='"en_US" "/univention/oidc/?location=/univention/portal/"' \
    --set anonymous=TRUE \
    --set activated=TRUE \
    --set linkTarget=samewindow \
    --set icon="$(base64 /usr/share/univention-portal/login.svg)"

$ udm portals/category modify --ignore_exists \
    --dn "cn=domain-service,cn=category,cn=portals,cn=univention,$(ucr get ldap/
↪base)"\
    --append entries="cn=login-oidc,cn=entry,cn=portals,cn=univention,$(ucr get␣
↪ldap/base)"
```

Verification and log files

To verify that the setup works, open the URL `https://FQDN/univention/oidc` in a web browser, such as Mozilla Firefox, and sign in. Open a UMC module, such as *Users*, and perform a search.

You find relevant logging information in the following locations:

- Log file: `/var/log/univention/management-console.server.log`
- **journald**: **`journalctl -u slapd.service`**

To reflect the changes for the login method in the portal, you need to edit the *Login* tile manually, similar to the setup with *SAML for single sign-on* (page 59). The link must point to `/univention/oidc/`.

Deactivate

First, you need to decide on which UCS systems you want to deactivate single sign-on using OpenID Connect. Second, you need to apply the following steps to each of those UCS systems.

1. Unset the Univention Configuration Registry Variable *umc/web/oidc/enabled* (page 306) with *ucr unset* (page 159):

```
$ ucr unset umc/web/oidc/enabled
```

2. Remove the OIDC RP[21] from Keycloak with the following command:

```
$ univention-keycloak oidc/rp remove \
   "$(ucr get umc/oidc/$(hostname -f)/client-id)"
```

3. Unset all Univention Configuration Registry Variables that you can find with the following searches:

```
$ ucr search --brief --key ^umc/oidc
$ ucr search --brief --key ^ldap/server/sasl/oauthbearer
```

4. Remove the OIDC secret from the system and restart affected services:

```
$ rm -f \
   /etc/umc-oidc.secret \
   /usr/share/univention-management-console/oidc/http*
$ systemctl restart slapd univention-management-console-server
```

5. Manually update the portal tile for *Login*, so that the link points to `/univention/login/`.
6. Change the Univention Configuration Registry Variable *portal/auth-mode* (page 302) to `ucs` with *ucr set* (page 158) and restart the portal server.

Identity Provider with non-standard FQDN

By default, the FQDN for the **`Keycloak`** identity provider is `ucs-sso-ng.$domainname`. However, it's possible to configure a different FQDN for the identity provider. For more information, see Configuration of the identity provider[22] in *Univention Keycloak app documentation* [4].

If you have such a setup, you have to configure the identity provider for the OpenID Connect authentication in UMC on each UCS system. Change the Univention Configuration Registry Variable *umc/oidc/issuer* (page 306) to the FQDN of your **`Keycloak`** identity provider and run the join script of the UMC web server again, as shown in Listing 4.2.

[21] https://docs.software-univention.de/keycloak-app/latest/architecture.html#term-OIDC-RP
[22] https://docs.software-univention.de/keycloak-app/latest/use-cases.html#use-case-custom-fqdn-idp

Listing 4.2: Set non-standard FQDN for identity provider **Keycloak**

```
$ IDP="auth.extern.test"
$ ucr set umc/oidc/issuer="https://$IDP/realms/ucs"
$ univention-run-join-scripts --force \
   --run-scripts 92univention-management-console-web-server
```

Non-standard FQDN for the Univention Portal and UMC

By default, the UMC is available under the FQDN `$hostname.$domainname`. If you have a setup with a different FQDN for the UMC you have to change the Univention Configuration Registry Variable *umc/oidc/rp/server* (page 306) to the FQDN of the UMC, and run the join script of the UMC web server again, as shown in Listing 4.3.

Listing 4.3: Set non-standard FQDN for the portal and UMC

```
$ ucr set umc/oidc/rp/server="portal.extern.test"
$ univention-run-join-scripts --force \
   --run-scripts 92univention-management-console-web-server
$ systemctl restart slapd
```

Important: If you want to run multiple Portal/UMC servers behind a load balancer, you need to run these commands on all UCS systems.

Since all the systems use the same OIDC client in this setup, make sure that the file `/etc/umc-oidc.secret` has the same contents on each system and matches the client secret in **Keycloak** for that client.

Back-channel sign-out

If you use OIDC back-channel sign-out together with multiprocessing of the UMC, the UMC needs a database for session storage to handle the session logout correctly. You have enabled multiprocessing in UMC if the Univention Configuration Registry Variable `umc/http/processes` has a value greater than one (`> 1`).

If you have only one UMC server without UMC multiprocessing, you don't need to change the configuration.

To keep track of the sessions in the database for UMC, you need to configure the database connection string with the **univention-mangement-console-settings** script, as shown in Listing 4.5.

However, if the Univention Portal or UMC uses multiple UCS servers for load balancing, or if UMC has a configuration for multiprocessing, it's necessary to use a **PostgreSQL** database that all the UCS systems can access. In these cases, you must consider the following aspects:

1. PostgreSQL database server:

   You either need to provide a **PostgreSQL** database yourself that all the UMC servers have access to.

   Or you install and configure **PostgreSQL** on one of the UCS servers. As shown in the example in Listing 4.4, you can freely choose the values for `db_user`, `db_name`, and `db_password`. `db_host` is the UCS system with **PostgreSQL** running.

Listing 4.4: Example for installation of **PostgreSQL**

```
$ univention-install univention-postgresql
$ su postgres -c "createdb db_name"
$ su postgres -c "/usr/bin/createuser db_user"
$ su postgres -c "psql db_name -c \"ALTER ROLE db_user WITH ENCRYPTED PASSWORD
↪'db_password'\""
```

(continues on next page)

(continued from previous page)

```
$ ucr set postgres11/pg_hba/config/host="db_name db_user 0.0.0.0/0 md5"
$ systemctl restart postgresql
```

2. Set the SQL connection URI on the Primary Directory Node:

Listing 4.5: Set SQL connection URI

```
$ univention-management-console-settings set \
   -u 'postgresql+psycopg2://db_user:db_password@db_host:5432/db_name'
```

3. Optional parameters for the database connection pool:

   - `Pool Size`: The number of connections to the database. The default value is `5`.
   - `Max Overflow`: The maximum number of temporary connections. The default value is `10`.
   - `Pool Timeout`: The number of seconds to wait for a connection to be available. The default value is `30`.
   - `Pool Recycle`: The number of seconds after which a connection is recycled. The default value is `-1`.

   With these default values, each UMC process can have up to 15 connections to the database. The total number of connections is: `<Number of Servers> * <Number of Processes> * (<Pool Size> + <Max Overflow>)`. Make sure that the database can handle the number of connections.

Listing 4.6: Set optional parameters for the database connection pool

```
$ univention-management-console-settings set \
     -s 5 \
     -o 10 \
     -t 30 \
     -r 3600
```

4. Restart the UMC on all UCS servers:

```
$ systemctl restart univention-management-console-server
```

Important: The feature for the refresh of the portal tabs on sign-out or session timeout requires **`PostgreSQL`**.

You can also use a local **`SQLite`** database for one UMC server with multiprocessing, or you can use **`MariaDB`** as central database for multiple load-balancing UMC servers. In both cases, the refresh of the portal tabs isn't supported and won't work, as it requires a **`PostgreSQL`** database.

4.3 UCS portal page

Portal pages offer a central view of all available services in a UCS domain. Requirements strongly differ from small to large environments in organizations, public authorities, or even schools. Therefore, UCS implemented a very flexible and individually customizable concept for portal pages.

Note: The Univention Portal is part of Univention Nubus. For more information about Nubus, refer to *What is Univention Nubus?* (page 2)

As illustrated in Fig. 4.4, portal entries (i.e., links to applications/Apps/services; UDM object type `portals/entry`) can be assigned to none, one or multiple portal categories. A portal category (UDM object type `portals/category`) can be assigned to none, one or multiple portals. A portal itself (UDM object type `portals/portal`) renders all portal categories which are assigned to it.

In addition, you can create portal announcements (UDM object type `portals/announcement`) and use them to announce, for example, maintenance windows or service outages. They can have a severity level, such as *information* or *warning*, a start and end date and a visibility limitation to a specific group of users. A portal announcement shows up at the top of each portal on all portals.

The portal *domain*, shipped with every installation, is configured on each server by default. In addition to all installed applications of the domain, links to Univention Management Console as well as the server overview are shown on this portal page.

To change the displayed portal, adjust the Univention Configuration Registry Variable *portal/default-dn* (page 303) and run the command **univention-portal update**.

Custom portals and portal entries can be defined and managed either via the UMC module *Portal* or directly on the portal site.

After logging in to the portal on the Primary Directory Node or Backup Directory Node, members of the `Domain Admins` group can edit the portal after clicking on the corresponding entry in the user menu. They now can create new entries on the portal, modify existing entries, modify the order or the design.

You can use the UMC module *Portal* for advanced settings such as adding portals and announcements or control through the group membership, which users can see which portal entries.

By default, all portal entries are displayed for everyone. In the UMC module *Portal* in the category *Login*, it can be configured whether anonymous visitors have to sign in before they can see entries. It is also possible to limit certain entries for certain groups. This requires the LDAP attribute `memberOf`. Nested group memberships (i.e., groups in groups) are evaluated.

You can make further design adjustments in the file `/usr/share/univention-portal/css/custom.css`. UCS doesn't overwrite this file during an update.

4.3.1 Assign rights for portal settings

The following describes how to make the UMC module *Portal* accessible to selected groups or users. This example assumes that a group `Portal Admins` has been created and members of this group are supposed to be given access to the portal settings.

On a Primary Directory Node an ACL file has to be created first, for example `/opt/62my-portal-acl.acl`. This file has to have the following content to allow the necessary ACL changes:

```
access to dn="cn=portals,cn=univention,@%@ldap/base@%@" attrs=children
  by group/univentionGroup/uniqueMember="cn=Portal Admins,cn=groups,@%@ldap/base@%@
↪" write
  by * +0 break

access to dn.children="cn=portals,cn=univention,@%@ldap/base@%@" attrs=entry,
↪@univentionObject,@univentionNewPortalEntry,
@univentionNewPortal,@univentionNewPortalCategory,children
  by group/univentionGroup/uniqueMember="cn=Portal Admins,cn=groups,@%@ldap/base@%@
↪" write
  by * +0 break
```

Then execute the following command to create an LDAP object for the LDAP ACLs:

```
$ udm settings/ldapacl create \
  --position "cn=ldapacl,cn=univention,$(ucr get ldap/base)" \
  --set name=62my-portal-acl \
  --set filename=62my-portal-acl \
  --set data="$(bzip2 -c /opt/62my-portal-acl.acl | base64)" \
  --set package="62my-portal-acl" \
  --set ucsversionstart=4.4-0 \
  --set ucsversionend=5.99-0 \
  --set packageversion=1
```

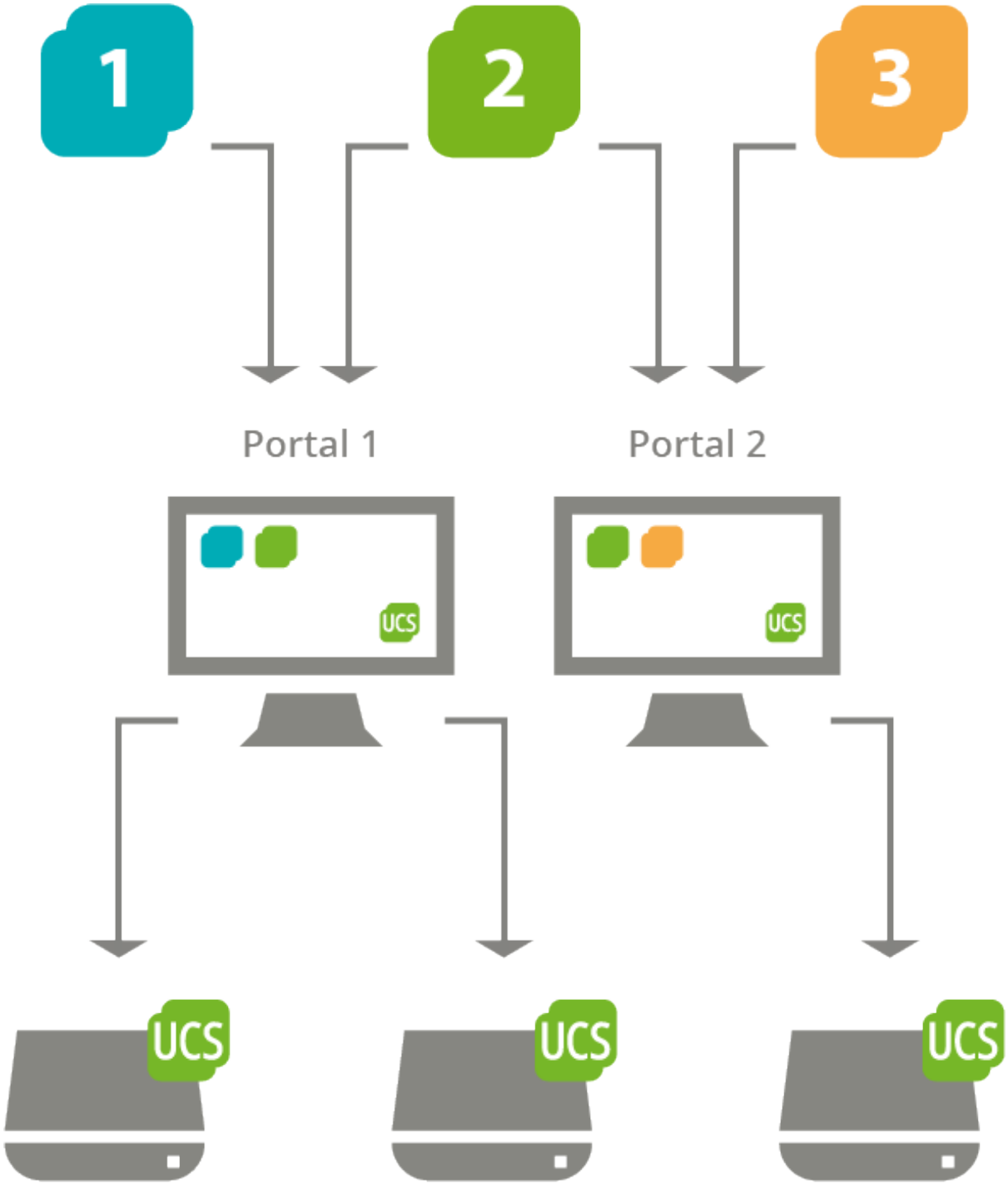


Fig. 4.4: Schema of the portal concept in UCS: Portals can be independently defined and assigned to UCS systems as start site; a link entry can be displayed on multiple portals.

If the ACL is to be deleted again, the following command can be used:

```
udm settings/ldapacl remove \
  --dn "cn=62my-portal-acl,cn=ldapacl,cn=univention,$(ucr get ldap/base)"
```

An appropriate UMC policy can now be created via UMC. The following *UMC operations* must be allowed within the policy:

- *udm-new-portal*
- *udm-syntax*
- *udm-validate*
- *udm-license*

How to create a policy is described in *Creating a policy* (page 75). Now the newly created policy only needs to be assigned to the wanted object, in this case the group `Portal Admins`. This can also be done directly within the UMC. For this example, navigate to the group module and edit the wanted group there. In the group settings, existing policies for the group object can be selected under *Policies*. More detailed information about policy assignment is described under *Applying policies* (page 75).

4.4 Consent for using Cookies

Both the UCS Portal and the Univention Management Console (UMC) use cookies and store them on the user's computer. Depending on your use case and the public presence of your UCS Portal, you may have to inform the user about the use of cookies.

When enabled, both the UCS Portal and the UMC show a cookie consent banner that the user must accept to continue. You can configure the content using Univention Configuration Registry Variables.

4.4.1 Usage of the cookie consent banner

To enable and customize the content of the cookie consent banner, use the following steps:

1. To enable the cookie consent banner, set the UCR variable *umc/cookie-banner/show* (page 69) to `true`. The banner then shows the default content.
2. To customize the title and the text, set the UCR variables *umc/cookie-banner/title* (page 69) and *umc/cookie-banner/text* (page 69) to your liking.

   Note that both settings allow a language-specific configuration. For more information, see the section *UCR reference for cookie consent banner* (page 69).
3. You can also optionally set the following variables:
   - The name of the cookie as UCS stores it on the user's system with *umc/cookie-banner/cookie* (page 69).
   - The domains for which UCS shows the cookie consent banner with *umc/cookie-banner/domains* (page 69).

Setting the appropriate UCR variables is sufficient to enable and customize the cookie consent banner.

To restore the default texts, unset the *umc/cookie-banner/title* (page 69) and *umc/cookie-banner/text* (page 69) UCR variables. To turn off the cookie banner completely, set *umc/cookie-banner/show* (page 69) to `false`.

4.4.2 UCR reference for cookie consent banner

Use the following UCR variables to configure the cookie banner dialog.

umc/cookie-banner/cookie

The variable sets the name of the cookie. In the default setting, the value is empty. UCS then uses the name `univentionCookieSettingsAccepted` for the cookie.

umc/cookie-banner/domains

Optional setting for the domains for which the cookie consent banner dialog is active. The value is a comma-separated list of domain names, for which UCS shows the cookie consent banner. For an empty list, UCS shows the banner for all domain names. The domain matches from the end of the string.

Examples:

- The value `example.com` matches `portal.example.com` and `sso.example com`. UCS shows the banner for both domain names.
- For the value `portal.example.com` UCS doesn't show the cookie consent banner for `sso.example.com`, but for `portal.example.com`.

umc/cookie-banner/show

The variable controls, if the browser shows the cookie consent banner. The default value is `false`. To show the cookie consent banner, set the variable to `true`.

umc/cookie-banner/title

Sets the title for the consent banner dialog. In the default setting, the value is empty and UCS provides a default title for English and German. Use `umc/cookie-banner/title/`*`LANGUAGE`* with a two letter language code from ISO 639-1[23] for *`LANGUAGE`* to set titles for different languages.

umc/cookie-banner/text

Sets the text for the cookie consent banner dialog. In the default setting, the value is empty and UCS provides a default text for English and German. Use `umc/cookie-banner/text/`*`LANGUAGE`* with a two letter language code from ISO 639-1[24] for *`LANGUAGE`* to set text content for different languages.

4.5 Univention Management Console modules

Univention Management Console (UMC) modules are the web-based tool for administration of the UCS domain. They are shown on the portal page (*UCS portal page* (page 65)) for logged in administrators. Depending on the system role, different UMC modules are available. Additionally installed software components may bring their own new UMC modules.

UMC modules for the administration of all the data included in the LDAP directory (such as users, groups and computer accounts) are only provided on Primary Directory Nodes and Backup Directory Node s. Changes made in these modules are applied to the whole domain.

UMC modules for the configuration and administration of the local system are provided on all system roles. These modules can for example be used to install additional applications and updates, adapt the local configuration via Univention Configuration Registry or start/stop services.

[23] https://en.wikipedia.org/wiki/ISO_639-1
[24] https://en.wikipedia.org/wiki/ISO_639-1

4.5.1 Activation of UCS license / license overview

The UCS license of a domain can be managed on the Primary Directory Node via the UMC module *Welcome!*.

The current license status can be shown by clicking the *License info* button.

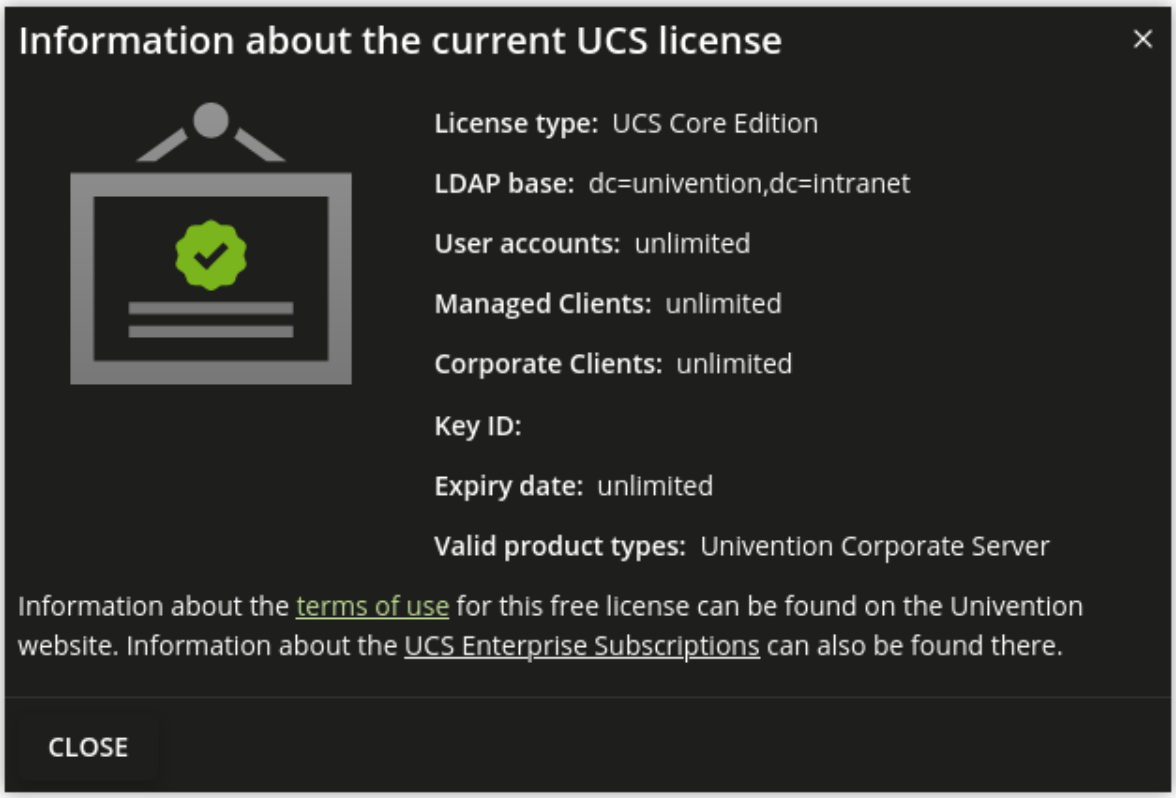


Fig. 4.5: Displaying the UCS license

The button *Import a license* opens a dialogue in which a new license key can be activated (otherwise the core edition license is used as default license). A license file can be selected and imported via the button *Import from file…*. Alternatively, the license key can also be copied into the input field below and activated with *Import from text field*.

Installation of most of the applications in the Univention App Center requires a personalized license key. UCS core edition licenses can be converted by clicking *Request a new license*. The current license key is sent to Univention and the updated key returned to a specified email address within a few minutes. The new key can be imported directly. The conversion does not affect the scope of the license.

If the number of licensed user or computer objects is exceeded, it is not possible to create any additional objects in UMC modules or edit any existing ones unless an extended license is imported or no longer required users or computers are deleted. A corresponding message is displayed when opening a UMC module if the license is exceeded.

4.5.2 Operating instructions for modules to administrate LDAP directory data

All UMC modules for managing LDAP directory objects such as user, group and computer accounts or configurations for printers, shares, mail and policies are controlled identically from a structural perspective. The following examples are presented using the user management but apply equally for all modules. The operation of the DNS and DHCP modules is slightly different. Further information can be found in *Administration of DNS data via Univention Management Console module* (page 214) and *Composition of the DHCP configuration via DHCP LDAP objects* (page 220).

The configuration properties/possibilities of the modules are described in the following chapters:

- Users - *User management* (page 105)

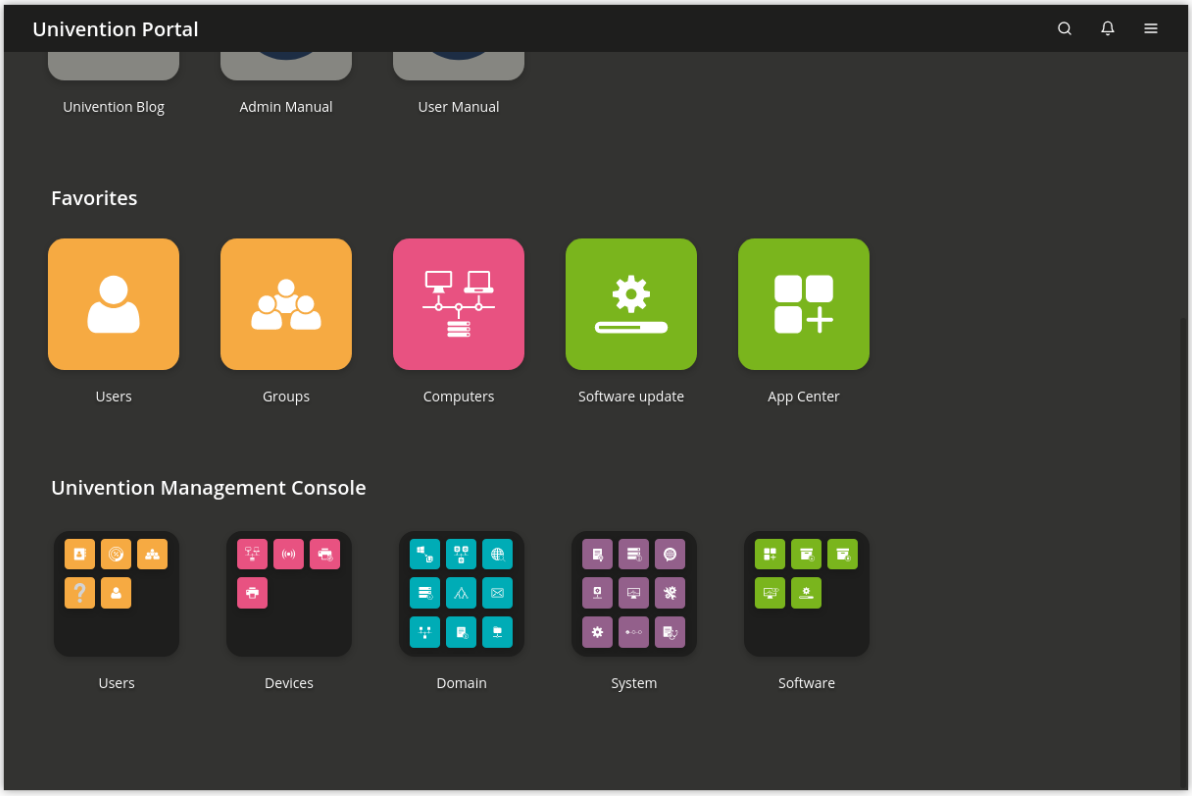


Fig. 4.6: Module overview

- Groups - *Group management* (page 135)
- Computers - *Computer management* (page 143)
- Networks - *IP and network management* (page 211)
- DNS - *Administration of DNS data with BIND* (page 212)
- DHCP - *IP assignment via DHCP* (page 220)
- Shares - *File share management* (page 237)
- Printers - *Print services* (page 249)
- Email - *Mail services* (page 261)
- Nagios - *Nagios* (page 287)

The use of policies (*Policies* (page 74)) and the LDAP navigation (*LDAP directory browser* (page 73)) are described separately.

Searching for objects

The module overview lists all the objects managed by this module. *Search* performs a search for a selection of important attributes (e.g., for user objects by first and last name, primary email address, description, employee number and username). A wildcard search is also possible, e.g., `m*`.

Clicking on the *Advanced options* button (the filter icon) next to the input field displays additional search options:

- The *Search in* field can be used to select whether the complete LDAP directory or only individual LDAP containers/OUs are searched. Further information on the structure of the LDAP directory service can be found in *Structuring of the domain with user-defined LDAP structures* (page 80).
- The *Property* field can be used to search for a certain attribute directly.

- The majority of the modules administrate a range of types of LDAP objects; the computer management for example administrates different objects for the individual system roles. The search can be limited to one type of LDAP object.
- Some of the internally used user groups and groups (e.g., for domain joins) are not shown by default. If the *Include hidden objects* option is enabled, these objects are also shown.

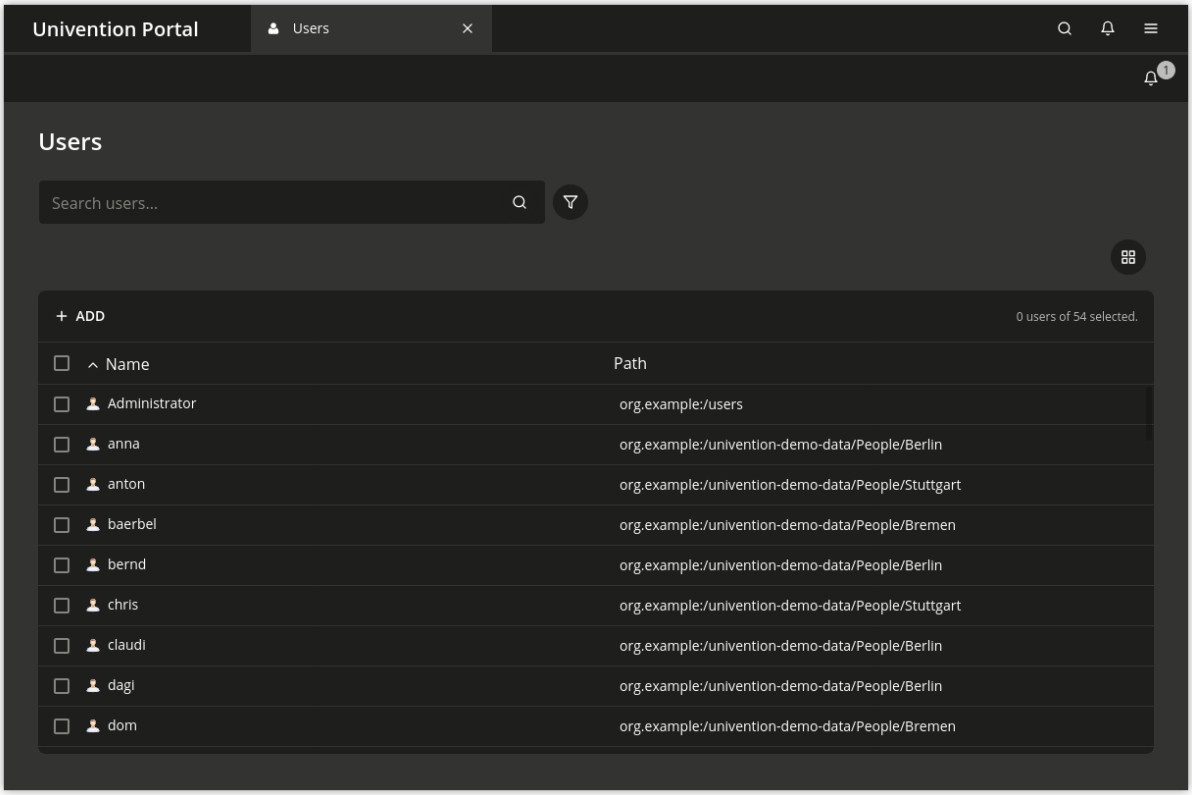


Fig. 4.7: Searching for users

Creating objects

At the top of the table that shows the objects is a toolbar which can be used to create a new object using *Add*.

There are simplified wizards for some UMC modules (users, hosts), in which only the most important settings are requested. All attributes can be shown by clicking on *Advanced*.

Editing objects

Right-clicking on an LDAP object and selecting *Edit* allows to edit the object. The individual attributes are described in the individual documentation chapters. By clicking on *Save* at the top of the module, all changes are written into the LDAP directory. The *Back* button cancels the editing and returns to the previous search view.

In front of every item in the result list is a checkbox with which individual objects can be selected. The selection status is also displayed in toolbar at the top of the table, e.g., *2 users of 102 selected*. If more than one object is selected, clicking on the *Edit* button in the toolbar activates the multi edit mode. The same attributes are now shown as when editing an individual object, but the changes are only accepted for the objects where the *Overwrite* checkbox is activated. Only objects of the same type can be edited at the same time.

Deleting objects

Right-clicking on an LDAP object and selecting *Delete* allows to delete the object. The prompt must be confirmed. Some objects use internal references (e.g., a DNS or DHCP object can be associated with computer objects). These can also be deleted by selecting the *Delete referring objects* option.

Similar to editing multiple objects at once, multiple objects can be deleted at once via the *Delete* button in the toolbar.

Moving objects

Right-clicking on an LDAP object and selecting *Move to…* allows to to select an LDAP position to which the object should be moved.

Similar to editing multiple objects at once, multiple objects can be moved at once by selecting *More* ‣ *Move to…* in the toolbar.

4.5.3 Display of system notifications

UMC modules can deploy system notifications to alert the user to potential errors like join scripts which have not been run or necessary actions such as available updates. These notifications are shown in the top right corner of the screen and can be viewed again in the Notifications menu, which can be opened by clicking the bell icon in the top right corner of the screen.

4.6 LDAP directory browser

The UMC module *LDAP directory* can be used to navigate through the LDAP directory. When doing so, new objects can be created, modified or deleted in the LDAP directory.

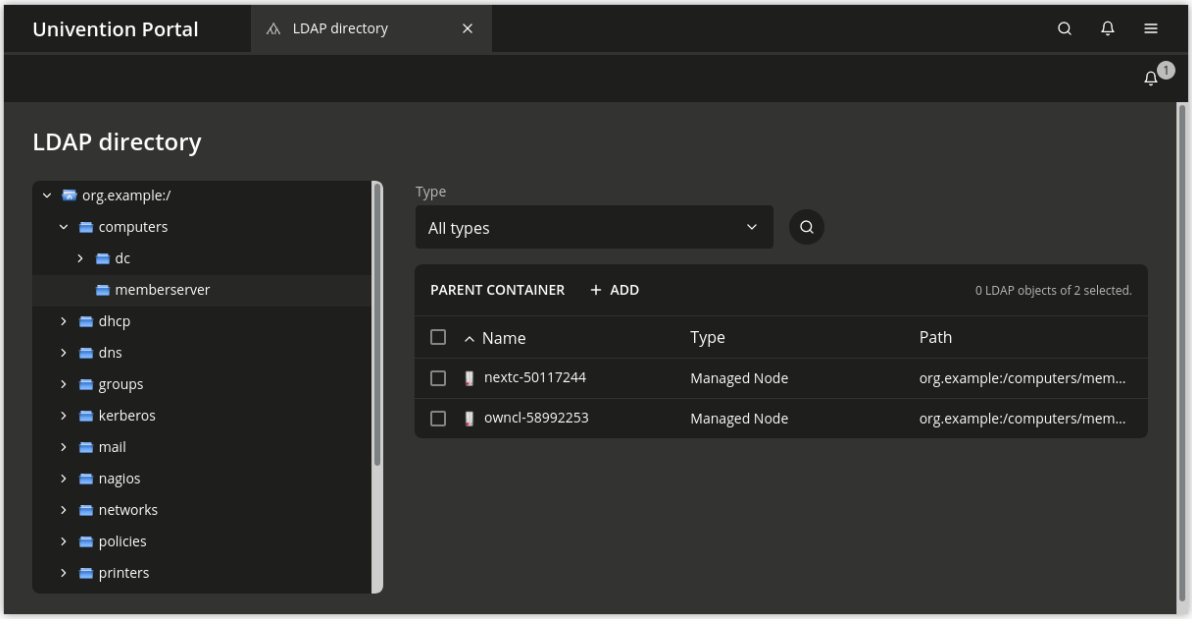


Fig. 4.8: Navigating the LDAP directory

The left half of the screen shows the LDAP directory as a tree structure whose elements can be shown and hidden using the arrow icons.

Clicking on an element of the tree structure switches to this LDAP position and displays the objects at this LDAP position in the in the right side of the screen. The *Type* selection list can be used to limit the display to selected attributes.

The *Add* button can be used to add new objects here too. Similar to the control elements described in *Univention Management Console modules* (page 69), existing objects can also be edited, deleted or moved here.

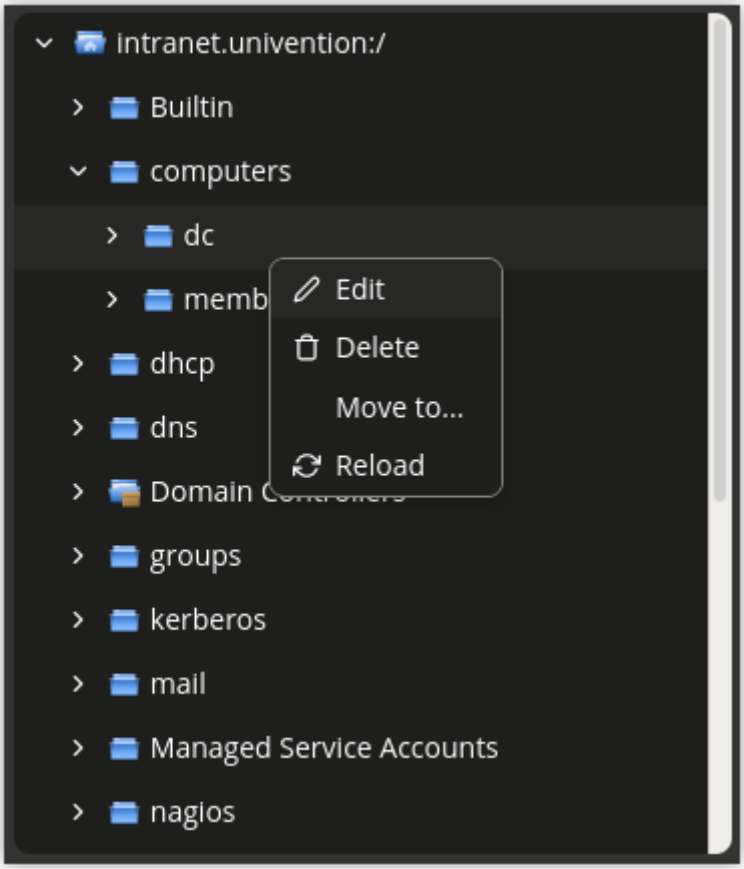


Fig. 4.9: Editing LDAP container settings

Right-clicking on an element in the tree structure allows editing the properties of the container or the LDAP base with *Edit*.

4.7 Policies

Policies describe administrative settings which can practically be used on more than one object. They facilitate the administration as they can be connected to containers and then apply to all the objects in the container in question and the objects in sub containers. The values are applied according to the inheritance principle. For every object, the applied value is always that which lies closest to the object in question.

If, for example, the same password expiry interval is to be defined for all users of a location, then a special container can be created for these users. After moving the user objects into the container, a password policy can be linked to the container. This policy is valid for all user objects within the container.

An exception to this rule is a value which was defined in a policy in the form of *fixed attributes*. Such values cannot be overwritten by subordinate policies.

The command line program **`univention-policy-result`** can be used to show in detail which policy applies to which directory service object.

Every policy applies to a certain type of UMC domain object, e.g., for users or DHCP subnets.

4.7.1 Creating a policy

Policies can be managed via the UMC module *Policies*. The operation is the same as for the functions described in *Univention Management Console modules* (page 69).

The attributes and properties of the policies are described in the corresponding chapters, e.g. the DHCP policies in the network chapter.

The names of policies must not contain any umlauts.

Referencing objects provides a list of all containers or LDAP objects for which this policy currently applies.

The expanded settings host some general policy options which are generally only required in special cases.

LDAP filter
A LDAP filter expression can be specified here, which an object must match for this policy to get applied.

Required object classes
Here you can specify LDAP object classes that an object must possess for the policy to apply to this object. If, for example, a user policy is only relevant for Windows environments, the `sambaSamAccount` object class could be demanded here.

Excluded object classes
Similar to the configuration of the required object classes, you can also list object classes here which should be excluded.

Fixed attributes
Attributes can be selected here, the values of which may not be changed by subordinate policies.

Empty attributes
Attributes can be selected here, which are to be set to empty in the policy, meaning they will be stored without containing a value. This can be useful for removing values inherited by an object from a superordinate policy. In subordinate policies, new values can be assigned to the attributes in question.

4.7.2 Applying policies

Policies can be assigned in two ways:

- A policy can be assigned to the LDAP base or a container/OU. To do so, the *Policies* tab in the properties of the LDAP object must be opened in the navigation (see *LDAP directory browser* (page 73)).
- A *Policies* tab is shown in the UMC modules of LDAP directory objects for which there are policies available (e.g., for users). A particular policy for a user can be specified at this place.

The *Policies* configuration dialogue is functionally identical; however, all policy types are offered when assigning policies to a LDAP container, whilst only the policy types applicable for the object type in question are offered when assigning policies to an LDAP object.

A policy can be assigned to the LDAP object or container under *Policies*. The values resulting from this policy are displayed directly. The *Inherited* setting means that the settings are adopted from a superordinate policy again - when one exists.

If an object is linked to a policy, or inherits policy settings which cannot be applied to the object, the settings remain without effect for the object. This makes it possible, for example, to assign a policy to the base entry of the LDAP directory, which is then valid for all the objects of the domain which can apply this policy. Objects which cannot apply to this policy are not affected.

4.7.3 Editing a policy

Policies can be edited and deleted in the UMC module *Policies*. The interface is described in *Univention Management Console modules* (page 69).

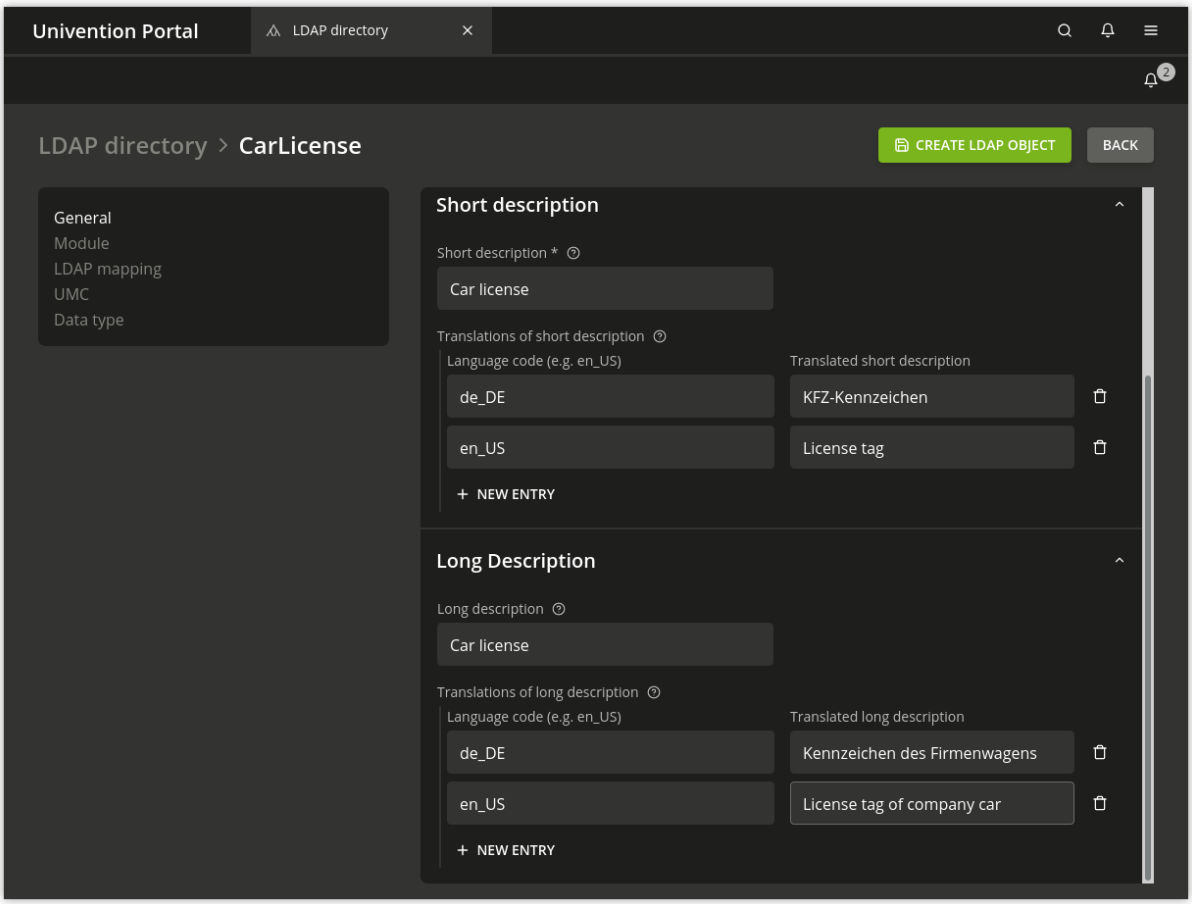
Caution: When editing a policy, the settings for all the objects linked to this policy are changed! The values from the changed policy apply to objects already registered in the system and linked to the policy, in the same way as to objects added in the future.

The policy tab of the individual LDAP objects also includes the *edit* option, which can be used to edit the policy currently applicable for this object.

4.8 Expansion of UMC modules with extended attributes

The domain management UMC modules allow the comprehensive management of the data in a domain. *Extended attributes* offer the possibility of integrating new attributes in the domain management which are not covered by the UCS standard scope. Extended attributes are also employed by third party vendors for the integration of solutions in UCS.

Extended attributes are managed in the UMC module *LDAP directory*. There one needs to switch to the `univention` container and then to the `custom attributes` subcontainer. Existing attributes can be edited here or a new *Settings: extended attribute* object created here with *Add*.



Extended attributes can be internationalized. In this case, the name and description should be compiled in English as this is the standard language for UMC modules.

4.8.1 Extended attributes - General tab

Table 4.1: *General* tab

Attribute	Description
Unique name	The name of the LDAP object which will be used to store the extended attribute. Within a container, the name has to be unique.
UDM CLI name	The specified attribute name should be used when employing the command line interface Univention Directory Manager. When the extended attribute is saved, the *Unique name* of the *General* tab is automatically adopted and can be subsequently modified.
Short description	Used as title of the input field in UMC modules or as the attribute description in the command line interface.
Translations of short description	Translated short descriptions can be saved in several languages so that the title of extended attributes is also output with other language settings in the respective national language. This can be done by assigning the respective short description to a language code (e.g., `de_DE` or `fr_FR`) in this input field.
Long description	This long description is shown as a tool tip in the input fields in UMC modules.
Translations of long description	Additional information displayed in the tool tip for an extended attribute can also be saved for several languages. This can be done by assigning the respective long description to a language code (e.g., `de_DE` or `fr_FR`) in this input field.

4.8.2 Extended attributes - Module tab

Table 4.2: *Module* tab

Attribute	Description
Modules to be extended	The Univention Directory Manager module which is to be expanded with the extended attribute. An extended attribute can apply for multiple modules.
Required options/object classes	Some extended attributes can only be used practically if certain object classes are activated on the *Options* tab. One or more options can optionally be saved in this input field so that this extended attribute is displayed or editable.
Hook class	The functions of the hook class specified here are used during saving, modifying and deleting the objects with extended attributes. Additional information can be found in *Univention Developer Reference* [3].

4.8.3 Extended attributes - LDAP mapping tab

Table 4.3: *LDAP mapping* tab

Attribute	Description
LDAP object class	Object class to which the attribute entered under *LDAP attribute* belongs. Predefined LDAP schema extensions for extended attributes are provided with the object class `univentionFreeAttributes`. Further information can be found in *LDAP schema extensions* (page 36). Each LDAP object which should be extended with an attribute is automatically extended with the LDAP object class specified here if a value for the extended attribute has been entered by the user.
LDAP attribute	The name of the LDAP attribute where the values of the LDAP object are to be stored. The LDAP attribute must be included in the specified object class.
Remove object class if the attribute is removed	If the value of an extended attribute in a UMC module is deleted, the attribute is removed from the LDAP object. If no further attributes of the registered object class are used in this LDAP object, the *LDAP object class* will also be removed from the LDAP object if this option is activated.

4.8.4 Extended attributes - UMC tab

Table 4.4: *UMC* tab

Attribute	Description
Do not show this extended attribute in UMC modules	This option can be activated if an attribute should only be administrated internally instead of by the administrator, e.g., indirectly by scripts. The attribute can then only be set via the command line interface Univention Directory Manager and is not displayed in UMC modules.
Exclude from UMC module	If it should not be possible to search for an extended attribute in the search window of a wizard, this option can be activated to remove the extended attribute from the list of possible search criteria. This is only needed in exceptional cases.
Ordering number	If several extended attributes are to be managed on one tab, the order of the individual attributes on the tab can be influenced here. They are added to the end of the tab or the group in question in ascending order of their numbers. Assigning consecutive position numbers results in the attributes being ordered on the left and right alternately in two columns. Otherwise, the positioning starts in the left column. If additional attributes have the same position number, their order is random.
Overwrite existing widget	In some cases it is useful to overwrite predefined input fields with extended attributes. If the internal UDM name of an attribute is configured here, its input field is overwritten by this extended attribute. The UDM attribute name can be identified with the command **`univention-directory-manager`** (see *Command line interface of domain management (Univention Directory Manager)* (page 81)). This option may cause problems if it is applied to a mandatory attribute.
Span both columns	As standard all input fields are grouped into two columns. This option can be used for overlong input fields, which need the full width of the tab.
Tab name	The name of the tab in UMC modules on which the extended attribute should be displayed. New tabs can also be added here. If no tab name is entered, *user-defined* will be used.
Translations of tab name	Translated tab names can be assigned to the corresponding language code (e.g. `de_DE` or `fr_FR`) in this input field.
Overwrite existing tab	If this option is activated, the tab in question is overwritten before the extended attributes are positioned on it. This option can be used to hide existing input fields on a predefined tab. It must be noted that this option can cause problems with compulsory fields. If the tab to be overwritten uses translations, the overwriting tab must also include identical translations.
Tab with advanced settings	Settings possibilities which are rarely used can be placed in the extended settings tab
Group name	Groups allow the structuring of a tab. A group is separated by a gray horizontal bar and can be shown and hidden. If no group name is specified for an extended attribute, the attribute is placed above the first group entry.
Translations of group name	To translate the name of the group, translated group names for the corresponding language code can be saved in this input field (e.g., `de_DE` or `fr_FR`).
Group ordering number	If multiple groups are managed in one tab, this position number can be used to specify the order of the groups. They are shown in the ascending order of their position numbers.

4.8.5 Extended attributes - Data type tab

Table 4.5: *Data type* tab

Attribute	Description
Syntax class	When values are entered in UMC modules, a syntax check is performed. Apart from standard syntax definitions (`string`) and (`integer`), there are three possibilities for expressing a binary condition. The syntax `TrueFalse` is represented at LDAP level using the strings `true` and `false`, the syntax `TrueFalseUpper` corresponds to the OpenLDAP boolean values `TRUE` and `FALSE` and the syntax `boolean` does not save any value or the string *1*. The syntax `string` is the default. An overview of the additionally available syntax definitions and instructions on integrating your own syntax can be found in *Univention Developer Reference* [3].
Default value	If a preset value is defined here, new objects to be created will be initialized with this value. The value can still be edited manually during creation. Existing objects remain unchanged.
Multi value	This option establishes whether a single value or multiple values can be entered in the input mask. The scheme definition of the LDAP attribute specifies whether one or several instances of the attribute may be used in one LDAP object.
Value required	If this option is active, a valid value must be entered for the extended attribute in order to create or save the object in question.
Editable after creation	This option establishes whether the object saved in the extended attribute can only be modified when saving the object, or whether it can also be modified subsequently.
Value is only managed internally	If this option is activated, the attribute cannot be modified manually, neither at creation time, nor later. This is useful for internal state information configured through a hook function or internally inside a module.
Copyable	Values of this extended attribute are automatically filled into the form when copying a object.

4.9 Structuring of the domain with user-defined LDAP structures

Containers and organizational units (OU) are used to structure the data in the LDAP directory. There is no technical difference between the two types, just in their application:

- Organizational units usually represent real, existing units such as a department in a company or an institution
- Containers are usually used for fictitious units such as all the computers within a company

Containers and organizational units are managed in the UMC module *LDAP directory* and are created with *Add* and the object types *Container: Container* and *Container: Organisational unit*.

Containers and OUs can in principle be added at any position in the LDAP; however, OUs cannot be created below containers.

4.9.1 General tab

Table 4.6: *General* tab

Attribute	Description
Name	A random name for the container / organizational unit.
Description	A random description for the container / organizational unit.

4.9.2 Advanced settings tab

Table 4.7: *Advanced settings* tab

Attribute	Description
Add to standard `[object type]` containers	If this option is activated, the container or organizational unit will be regarded as a standard container for a certain object type. If the current container is declared the standard user container, for example, this container will also be displayed in users search and create masks.

4.9.3 Policies tab

The *Policies* tab is described in *Applying policies* (page 75).

4.10 Delegated administration for UMC modules

By default only the members of the `Domain Admins` group can access all UMC modules. Policies can be used to configure the access to UMC modules for groups or individual users. For example, this can be used to assign a helpdesk team the authority to manage printers without giving them complete access to the administration of the domain.

UMC modules are assigned via a *UMC* policy which can be assigned to user and group objects. The evaluation is performed additively, i.e., general access rights can be assigned via ACLs assigned to groups and these rights can be extended via ACLs bound to user (see *Policies* (page 74)).

In addition to the assignment of UMC policies, LDAP access rights need to be taken into account, as well, for modules that manage data in the LDAP directory. All LDAP modifications are applied to the whole UCS domain. Therefore by default only members of the `Domain Admins` group and some internally used accounts have full access to the UCS LDAP. If a module is granted via a UMC policy, the LDAP access must also be allowed for the user/group in the LDAP ACLs. Further information on LDAP ACLs can be found in *Access control for the LDAP directory* (page 38).

Table 4.8: Policy *UMC*

Attribute	Description
List of allowed UCS operation sets	All the UMC modules defined here are displayed to the user or group to which this ACL is applied. The names of the domain modules begin with `UDM`.

Caution: For access to UMC modules, only policies are considered that are assigned to groups or directly to user and computer accounts. Nested group memberships (i.e., groups in groups) are not evaluated.

4.11 Command line interface of domain management (Univention Directory Manager)

The Univention Directory Manager is the command line interface alternative to the web-based interface of the domain management UMC modules. It functions as a powerful tool for the automation of administrative procedures in scripts and for the integration in other programs.

Univention Directory Manager can be started with the **`univention-directory-manager`** command (short form **`udm`**) as the `root` user on the Primary Directory Node.

UMC modules and Univention Directory Manager use the same domain management modules, i.e., all functions of the web interface are also available in the command line interface.

4.11.1 Parameters of the command line interface

A complete list of available modules is displayed if the **udm`** is run with the `modules` parameter:

```
$ univention-directory-manager modules
Available Modules are:
  computers/computer
  computers/domaincontroller_backup
  computers/domaincontroller_master
  computers/domaincontroller_slave
  [...]
```

There are up to five operations for every module:

list
: lists all existing objects of this type.

create
: creates a new object.

modify
: or the *editing* of existing objects.

remove
: deletes an object.

move
: is used to move an object to another position in the LDAP directory.

The possible options of a UDM module and the operations which can be used on it can be output by specifying the operation name, e.g.,

```
$ univention-directory-manager users/user move
[...]
general options:
  --binddn                          bind DN
  --bindpwd                         bind password
  --bindpwdfile                     file containing bind password
[...]
create options:
  --position                        Set position in tree
  --set                             Set variable to value, e.g. foo=bar
[...]
modify options:
  --dn                              Edit object with DN
  --set                             Set variable to value, e.g. foo=bar
[...]
remove options:
  --dn                              Remove object with DN
  --superordinate                   Use superordinate module
[...]
list options:
  --filter                          Lookup filter
  --position                        Search underneath of position in tree
[...]
move options:
  --dn                              Move object with DN
  --position                        Move to position in tree
[...]
```

The following command outputs further information, the operations and the options for every module. This also displays all attributes of the module:

```
$ univention-directory-manager [category/modulename]
```

With the `create` operation, the attributes marked with `*` must be specified when creating a new object.

Some attributes can be assigned more than one value (e.g., mail addresses to user objects). These multi-value fields are marked with `[]` behind the attribute name. Some attributes can only be set if certain options are set for the object. This is performed for the individual attributes by entering the option name:

```
users/user variables:
  General:
    username (*)                              Username
[...]
  Contact:
    e-mail (person,[])                        E-Mail Address
```

Here, `username (*)` signifies that this attribute must always be set when creating user objects. If the *person* option is set for the user account (this is the standard case), one or more email addresses can be added to the contact information.

A range of standard parameters are defined for every module:

`--dn`

The parameter is used to specify the LDAP position of the object during modifications or deletion. The complete DN must be entered, e.g.,

```
$ univention-directory-manager users/user remove \
  --dn "uid=ldapadmin,cn=users,dc=company,dc=example"
```

`--position`

The parameter is used to specify at which LDAP position an object should be created. If no `--position` is entered, the object is created below the LDAP base! In the `move` operation, this parameter specifies to which position an object should be moved, e.g:

```
$ univention-directory-manager computers/ipmanagedclient move \
  --dn "cn=desk01,cn=management,cn=computers,dc=company,dc=com" \
  --position "cn=finance,cn=computers,dc=company,dc=example"
```

`--set`

The parameter specifies that the given value should be assigned to the following attribute. The parameter must be used per attribute value pair, e.g:

```
$ univention-directory-manager users/user create \
  --position "cn=users,dc=compaby,dc=example" \
  --set username="jsmith" \
  --set firstname="John" \
  --set lastname="Smith" \
  --set password="12345678"
```

`--option`

The parameter defines the LDAP object classes of an object. If, for example, only `pki` is provided as options for a user object, it is not possible to specify a `mailPrimaryAddress` for this user as this attribute is part of the `mail` option.

`--superordinate`

`--superordinate` is used to specify dependent, superordinate modules. A DHCP object, for example, requires a DHCP service object under which it can be stored. This is transferred with the `--superordinate` option.

--policy-reference

The `--policy-reference` parameter allows the assignment of policies to objects (and similarly their deletion with `--policy-dereference`). If a policy is linked to an object, the settings from the policy are used for the object, e.g.:

```
$ univention-directory-manager [category | modulename] [Operation] \
  --policy-reference "cn=sales,cn=pwhistory," \
"cn=users,cn=policies,dc=company,dc=example"
```

--ignore-exists

The `--ignore_exists` parameters skips existing objects. If it is not possible to create an object, as it already exists, the error code `0` (no error) is still returned.

--append

`--append` and `--remove` are used to add/remove a value from a multi-value field, e.g.:

```
$ univention-directory-manager groups/group modify \
  --dn "cn=staff,cn=groups,dc=company,dc=example" \
  --append users="uid=smith,cn=users,dc=company,dc=example" \
  --remove users="uid=miller,cn=users,dc=company,dc=example"
```

--remove

See *--append* (page 84).

4.11.2 Example invocations of the command line interface

The following examples for the command line frontend of Univention Directory Manager can be used as templates for your own scripts.

Users

Creating a user in the standard user container:

```
$ univention-directory-manager users/user create \
  --position "cn=users,dc=example,dc=com" \
  --set username="user01" \
  --set firstname="Random" \
  --set lastname="User" \
  --set organisation="Example company LLC" \
  --set mailPrimaryAddress="mail@example.com" \
  --set password="secretpassword"
```

Subsequent addition of the postal address for an existing user:

```
$ univention-directory-manager users/user modify \
  --dn "uid=user01,cn=users,dc=example,dc=com" \
  --set street="Exemplary Road 42" \
  --set postcode="28239" \
  --set city="Bremen"
```

This command can be used to display all the users whose username begins with *user*:

```
$ univention-directory-manager users/user list \
  --filter uid='user*'
```

Searching for objects with the `--filter` can also be limited to a position in the LDAP directory; in this case, to all users in the container `cn=bremen,cn=users,dc=example,dc=com`:

```
$ univention-directory-manager users/user list \
  --filter uid="user*" \
  --position "cn=bremen,cn=users,dc=example,dc=com"
```

This call removes the user `user04`:

```
$ univention-directory-manager users/user remove \
  --dn "uid=user04,cn=users,dc=example,dc=com"
```

A company has two sites with containers created for each. The following command can be used to transfer a user from the container for the site “Hamburg” to the container for the site “Bremen”:

```
$ univention-directory-manager users/user move \
  --dn "uid=user03,cn=hamburg,cn=users,dc=example,dc=com" \
  --position "cn=bremen,cn=users,dc=example,dc=com"
```

Groups

Creating a group `Example Users` and adding the user `user01` to this group:

```
$ univention-directory-manager groups/group create \
  --position "cn=groups,dc=example,dc=com" \
  --set name="Example Users" \
  --set users="uid=user01,cn=users,dc=example,dc=com"
```

Subsequent addition of the user `user02` to the existing group:

```
$ univention-directory-manager groups/group modify \
  --dn "cn=Example Users,cn=groups,dc=example,dc=com" \
  --append users="uid=user02,cn=users,dc=example,dc=com"
```

Caution: A `--set` on the attribute `users` overwrites the list of group members in contrast to `--append`.

Subsequent removal of the user `user01` from the group:

```
$ univention-directory-manager groups/group modify \
  --dn "cn=Example Users,cn=groups,dc=example,dc=com" \
  --remove users="uid=user01,cn=users,dc=example,dc=com"
```

Container / Policies

This call creates a container `cn=Bremen` beneath the standard container `cn=computers` for the computers at the “Bremen” site. The additional option `computerPath` also registers this container directly as the standard container for computer objects (see *Structuring of the domain with user-defined LDAP structures* (page 80)):

```
$ univention-directory-manager container/cn create \
  --position "cn=computers,dc=example,dc=com" \
  --set name="bremen" \
  --set computerPath=1
```

This command creates a disk quota policy with soft and hard limits and the name *Default quota*:

```
$ univention-directory-manager policies/share_userquota create \
  --position "cn=policies,dc=example,dc=com" \
  --set name="Default quota" \
  --set softLimitSpace=5GB \
  --set hardLimitSpace=10GB
```

This policy is now linked to the user container `cn=users`:

```
$ univention-directory-manager container/cn modify \
  --dn "cn=users,dc=example,dc=com" \
  --policy-reference "cn=Default quota,cn=policies,dc=example,dc=com"
```

Creating a Univention Configuration Registry policy with which the storage time for log files can be set to one year. One space is used to separate the name and value of the variable:

```
$ univention-directory-manager policies/registry create \
  --position "cn=config-registry,cn=policies,dc=example,dc=com" \
  --set name="default UCR settings" \
  --set registry="logrotate/rotate/count 52"
```

This command can be used to attach an additional value to the created policy:

```
$ univention-directory-manager policies/registry modify \
  --dn "cn=default UCR settings,cn=config-registry,cn=policies,dc=example,dc=com" \
  --append registry='"logrotate/compress" "no"'
```

Computers

In the following example, a Windows client is created. If this client joins the Samba domain at a later point in time (see *Windows domain joins* (page 31)), this computer account is then automatically used:

```
$ univention-directory-manager computers/windows create \
  --position "cn=computers,dc=example,dc=com" \
  --set name=WinClient01 \
  --set mac=aa:bb:cc:aa:bb:cc \
  --set ip=192.0.2.10
```

Shares

The following command creates a share *Documentation* on the server *fileserver.example.com*. As long as `/var/shares/documentation/` does not yet exist on the server, it is also created automatically:

```
$ univention-directory-manager shares/share create \
  --position "cn=shares,dc=example,dc=com" \
  --set name="Documentation" \
  --set host="fileserver.example.com" \
  --set path="/var/shares/documentation"
```

Printers

Creating a printer share *LaserPrinter01* on the print server *printserver.example.com*. The properties of the printer are specified in the PPD file, the name of which is given relative to the directory `/usr/share/ppd/`. The connected printer is network-compatible and is connected via the IPP protocol.

```
$ univention-directory-manager shares/printer create \
  --position "cn=printers,dc=example,dc=com" \
  --set name="LaserPrinter01"  \
  --set spoolHost="printserver.example.com" \
  --set uri="ipp:// 192.0.2.100" \
  --set model="foomatic-rip/HP-Color_LaserJet_9500-Postscript.ppd" \
  --set location="Head office" \
  --set producer="producer: cn=HP,cn=cups,cn=univention,dc=example,dc=com"
```

Note: There must be a blank space between the print protocol and the URL target path in the parameter `uri`. A list of the print protocols can be found in *Creating a printer share* (page 250).

Printers can be grouped in a printer group for simpler administration. Further information on printer groups can be found in *Creating a printer group* (page 253).

```
$ univention-directory-manager shares/printergroup create \
  --set name=LaserPrinters \
  --set spoolHost="printserver.example.com" \
  --append groupMember=LaserPrinter01 \
  --append groupMember=LaserPrinter02
```

DNS/DHCP

To configure an IP assignment via DHCP, a DHCP computer entry must be registered for the MAC address. Further information on DHCP can be found in *IP assignment via DHCP* (page 220).

```
$ univention-directory-manager dhcp/host create \
  --superordinate "cn=example.com,cn=dhcp,dc=example,dc=com" \
  --set host="Client222" \
  --set fixedaddress="192.0.2.110" \
  --set hwaddress="ethernet 00:11:22:33:44:55"
```

If it should be possible for a computer name to be resolved via DNS, the following commands can be used to configure a forward (host record) and reverse resolution (PTR record).

```
$ univention-directory-manager dns/host_record create \
  --superordinate "zoneName=example.com,cn=dns,dc=example,dc=com" \
  --set name="Client222" \
  --set a="192.0.2.110"

$ univention-directory-manager dns/ptr_record create \
  --superordinate "zoneName=0.168.192.in-addr.arpa,cn=dns,dc=example,dc=com" \
  --set address="110" \
  --set ptr_record="Client222.example.com."
```

Further information on DNS can be found in *Administration of DNS data with BIND* (page 212).

Extended attributes

Extended attributes can be used to expand the functional scope of UMC modules, see *Expansion of UMC modules with extended attributes* (page 76). In the following example, a new attribute is added, where the car license number of the company car can be saved for each user. The values are managed in the object class `univentionFreeAttributes` created specially for this purpose:

```
$ univention-directory-manager settings/extended_attribute create \
  --position "cn=custom attributes,cn=univention,dc=example,dc=com" \
  --set name="CarLicense" \
  --set module="users/user" \
  --set ldapMapping="univentionFreeAttribute1" \
  --set objectClass="univentionFreeAttributes" \
  --set longDescription="License plate number of the company car" \
  --set tabName="Company car" \
  --set multivalue=0 \
  --set syntax="string" \
  --set shortDescription="Car license"
```

4.12 HTTP API of domain management

UCS provides an HTTP API for UDM which can be used to inspect, modify, create and delete UDM objects via HTTP requests.

For more information on the API please refer to *Univention Developer Reference* [3].

4.13 Evaluation of data from the LDAP directory with Univention Directory Reports

Univention Directory Reports offers the possibility of creating predefined reports for any objects to be managed in the directory service.

The structure of the reports is defined using templates. The specification language developed for this purpose allows the use of wildcards, which can be replaced with values from the LDAP directory. Any number of report templates can be created. This allows users to select very detailed reports or just create simple address lists, for example.

The creation of the reports is directly integrated in the UMC modules *Users*, *Groups* and *Computers*. Alternatively, the command line program **univention-directory-reports** can be used.

Six report templates are already provided with the delivered Univention Directory Reports, which can be used for users, groups and computers. Three templates create PDF documents and three CSV files, which can be used as an import source for other programs. Further templates can be created and registered.

4.13.1 Creating reports via Univention Management Console modules

To create a report, you need to open the UMC module *Users*, *Groups* or *Computers*. Then all the objects which should be covered by the report must be selected (you can select all objects by clicking the checkbox the left of *Name*). Clicking on *More* ▸ *Create report* allows to choose between the *Standard Report* in PDF format and the *Standard CSV Report* in CSV format.

The reports created via a UMC module are stored for 12 hours and then deleted by a cron job. The settings for when the cron job should run and how long the reports should be stored for can be defined via two Univention Configuration Registry variables:

directory/reports/cleanup/cron
: Defines when the cron job should be run.

directory/reports/cleanup/age
: Defines the maximum age of a report document in seconds before it is deleted.

4.13.2 Creating reports on the command line

Reports can also be created via the command line with the **univention-directory-reports** program. Information on the use of the program can be viewed using the `--help` option.

The following command can be used to list the report templates available to users, for example:

```
$ univention-directory-reports -m users/user -l
```

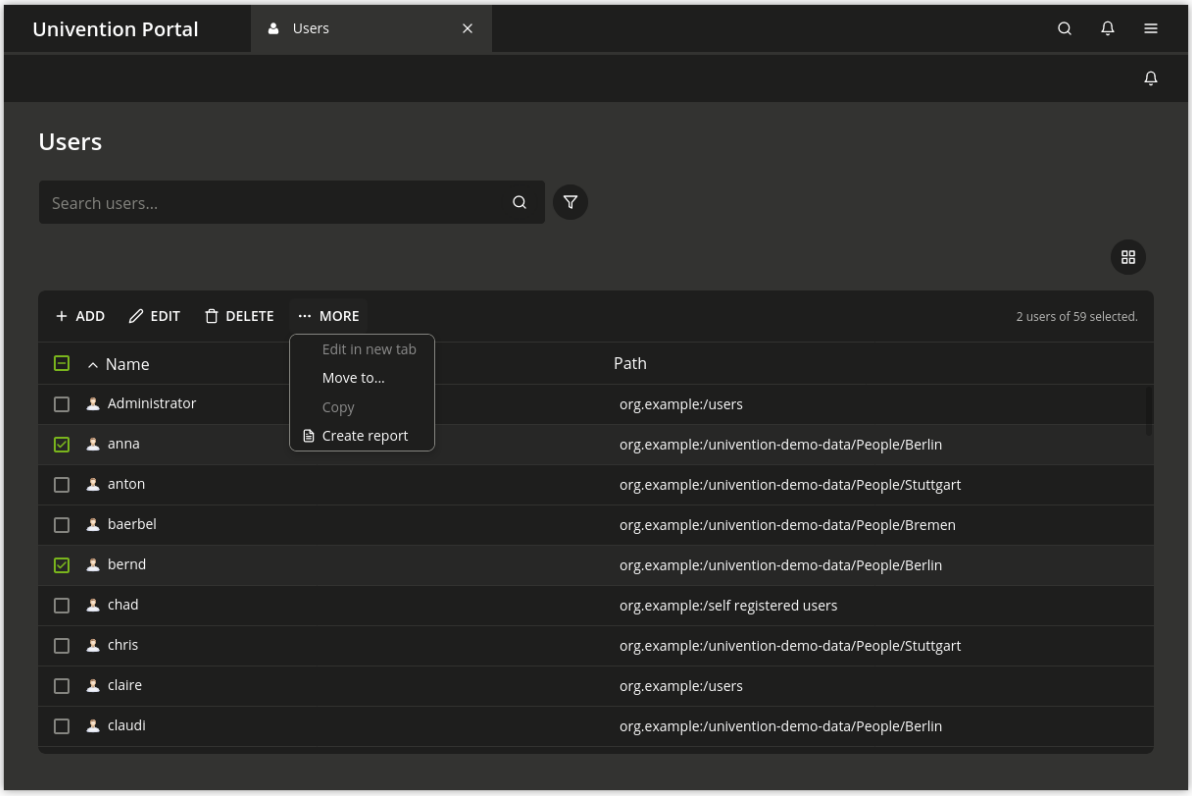


Fig. 4.10: Creating a report

4.13.3 Adjustment/expansion of Univention Directory Reports

Existing reports can be created directly with the presettings. Some presettings can be adapted using Univention Configuration Registry. For example, it is possible to replace the logo that appears in the header of each page of a PDF report. To do so, the value of the Univention Configuration Registry Variable `directory/reports/logo` (page 293) can include the name of an image file. The usual image formats such as JPEG, PNG and GIF can be used. The image is automatically adapted to a fixed width of 5.0 cm.

In addition to the logo, the contents of the report can also be adapted by defining new report templates.

4.14 Let's Encrypt

Let's Encrypt is a non-profit certificate authority that provides X.509 certificates for TLS encryption at no charge. It is the world's largest certificate authority with the goal of all websites being secure and using HTTPS.

The **`Let's Encrypt`** app in Univention App Center offers a largely automated integration of the *acme-tiny Let's Encrypt client* in UCS. The supported services in UCS are the Apache Web server, the Postfix SMTP mail server and the Dovecot IMAP mail server.

CHAPTER

FIVE

SOFTWARE DEPLOYMENT

The software deployment integrated in UCS offers extensive possibilities for the rollout and updating of UCS installations. Security and version updates can be installed via the UMC module *Software update*, a command line tool or based on policies. This is described in the section *Updates of UCS systems* (page 96). The UCS software deployment does not support the updating of Microsoft Windows systems. An additional Windows software distribution is required for this.

For larger installations, there is the possibility of establishing a local repository server from which all further updates can be performed, see *Configuration of the repository server for updates and package installations* (page 98).

The UCS software deployment is based on the underlying Debian package management tools, which are expanded through UCS-specific tools. The different tools for the installation of software are introduced in *Installation of further software* (page 99). The installation of version and errata updates can be automated via policies, see *Specification of an update point using the package maintenance policy* (page 102).

The software monitor provides a tool with which all package installations statuses can be centrally stored in a database, see *Central monitoring of software installation statuses with the software monitor* (page 103).

The initial installation of UCS systems is not covered in this chapter, but is documented in *Installation* (page 9) instead.

5.1 Differentiation of update variants / UCS versions

Four types of UCS updates are differentiated:

Major releases
: *Major releases* appear approximately every three to four years. Major releases can differ significantly from previous major releases in terms of their scope of services, functioning and the software they contain.

Minor releases
: During the maintenance period of a major release, *minor releases* are released approximately every 10-12 months. These updates include corrections to recently identified errors and the expansion of the product with additional features. At the same time and as far as this is possible, the minor releases are compatible with the previous versions in terms of their functioning, interfaces and operation. Should a change in behavior prove practical or unavoidable, this will be noted in the release notes when the new version is published.

Patchlevel releases
: *Patchlevel releases* are released approximately every three months and combine all errata updates published until then.

Errata updates
: Univention continuously releases *errata updates*. Errata updates provide fixes for security vulnerabilities, bug fixes, and smaller enhancements to make them available to customer systems quickly. An overview of all errata updates can be found at https://errata.software-univention.de/.

Every released UCS version has an unambiguous version number; it is composed of a figure (the major version), a full stop, a second figure (the minor version), a hyphen and a third figure (the patch level version). The version UCS 4.2-1 thus refers to the first patch level update for the second minor update for the major release UCS 4.

The *pre-update script* `preup.sh` is run before every release update. It checks for example whether any problems exist, in which case the update is canceled in a controlled manner. The *post-update script* `postup.sh` is run at the end of the update to perform additional cleanups, if necessary.

Errata updates always refer to certain minor releases, e.g., for UCS 5.0. Errata updates can generally be installed for all patch level versions of a minor release.

If a new release or errata updates are available, a corresponding notification is given when a user opens a UMC module. The availability of new updates is also notified via email; the corresponding newsletters - separated into release and error updates - can be subscribed on the Univention website. A changelog document is published for every release update listing the updated packages, information on error corrections and new functions and references to the Univention Bugzilla.

5.2 Univention App Center

The Univention App Center allows simple integration of software components in a UCS domain. The applications are provided both by third parties and by Univention itself (e.g., UCS@school). The maintenance and support for the applications are provided by the respective manufacturer.

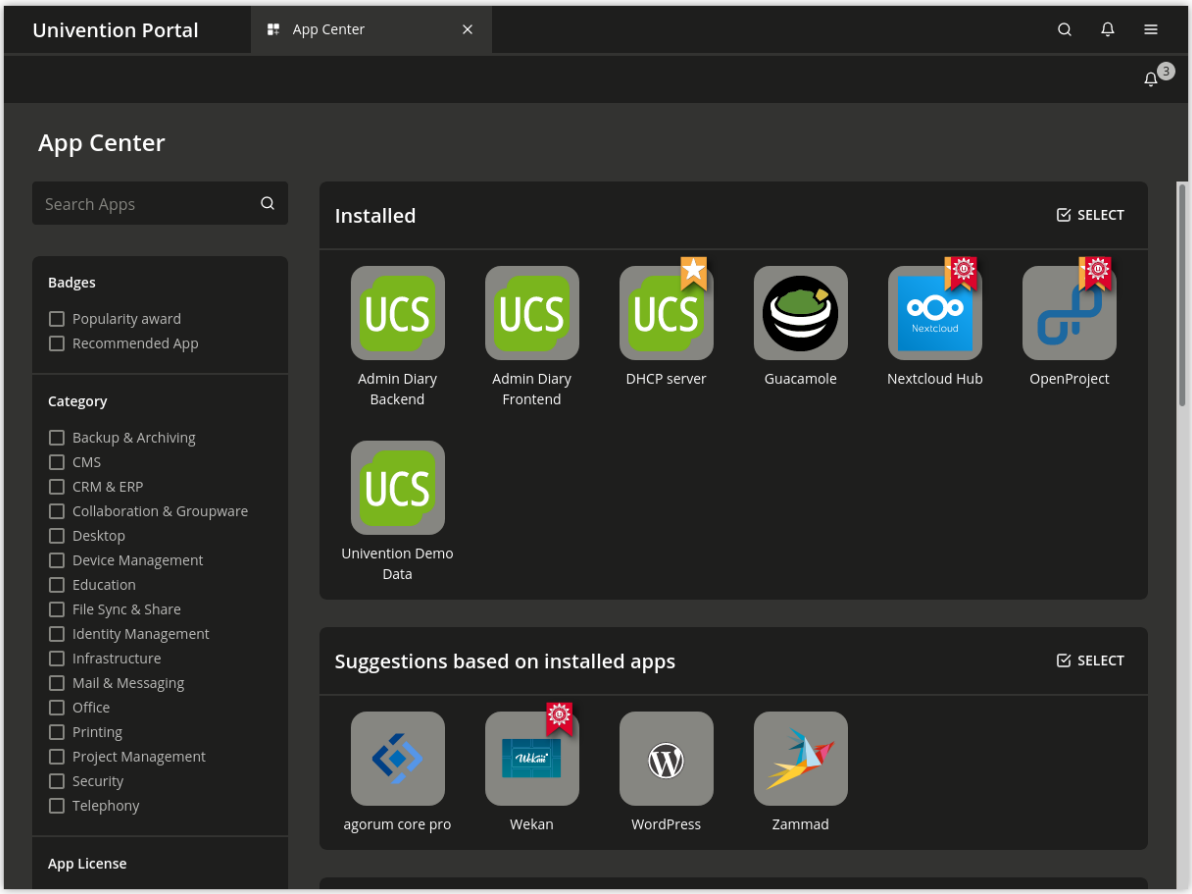


Fig. 5.1: Overview of applications available in the App Center

The Univention App Center can be opened via the UMC module *App Center*. It shows by default all installed as well as available software components. *Search Apps…* can be used to search for available applications. Furthermore, the applications can also be filtered using the *Category* panel. More filters like the *Badges* and the *App License* can be used. For example, the view can be limited to applications with the categories `Education` or `Office`. To only show the `Recommended Apps` for theses categories, it is sufficient to activate the appropriate filter.

If you click on one of the displayed applications, further details on it are shown, for example description, manufacturer, contact information and screenshots or videos. The *Notification* field displays whether the manufacturer of the

software component is notified when it is installed/uninstalled. A rough classification of the licensing can be found under the *License* section. Some applications provide a *Buy* button with a link to detailed licensing information. For all other applications, it is recommended to contact the manufacturer of the application about detailed licensing information using the email address shown under *Contact*.

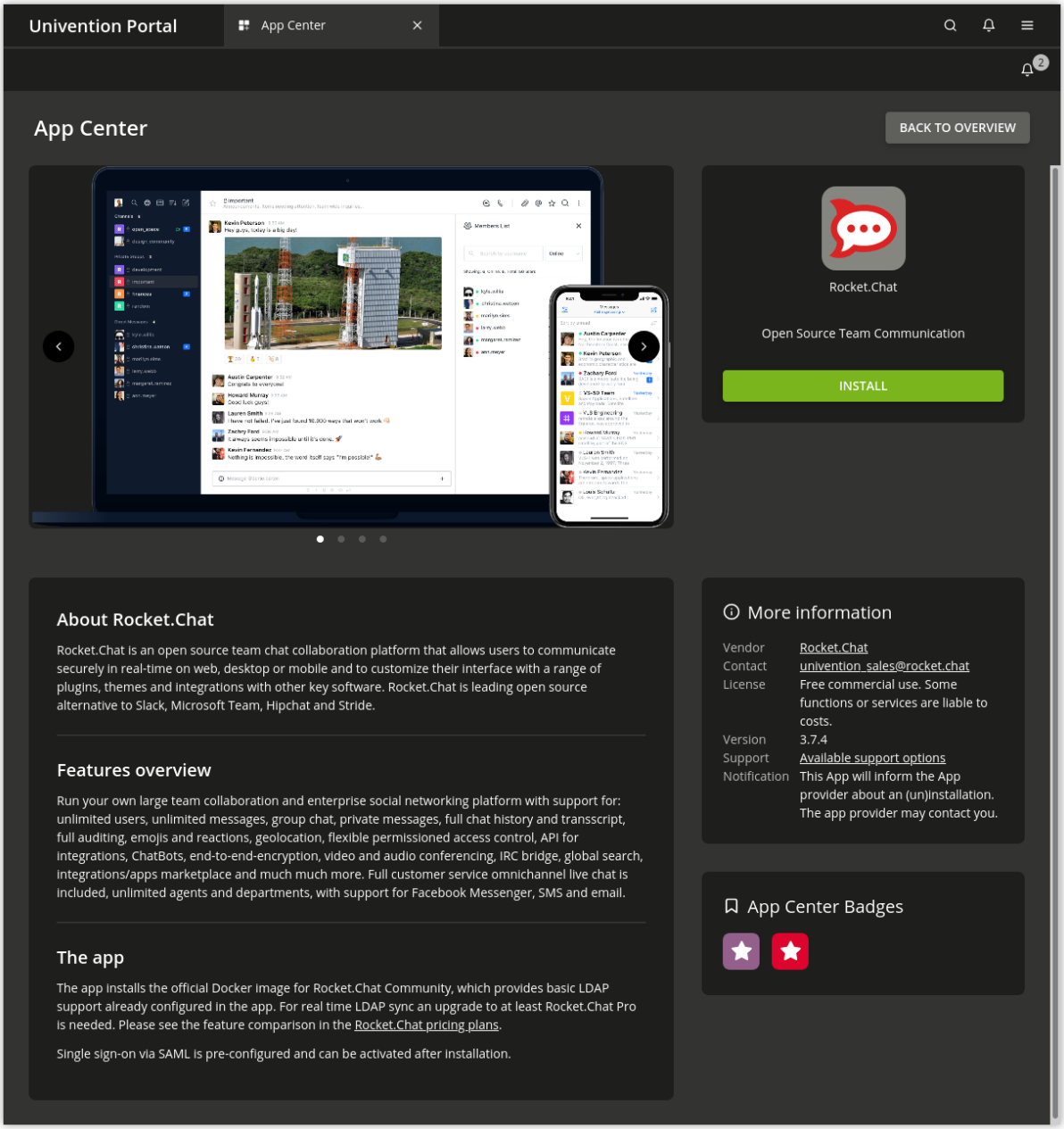


Fig. 5.2: Details for an application in the App Center

With *Vote Apps* there is a special form of Apps in the App Center that do not install anything on the UCS system. Voting helps Univention and the potential app provider to determine the interest in this app. Vote apps are usually only displayed for a limited voting period. That Vote Apps are available, can be recognized by the shown *Vote Apps* filter option in the App Center overview.

Some applications may not be compatible with other software packages from UCS. For instance, most groupware packages require the UCS mail stack to be uninstalled. Every application checks whether incompatible versions are installed and then prompts which *Conflicts* exist and how they can be resolved. The installation of these packages is then prevented until the conflicts have been resolved.

Some components integrate packages that need to be installed on the Primary Directory Node (usually LDAP schema

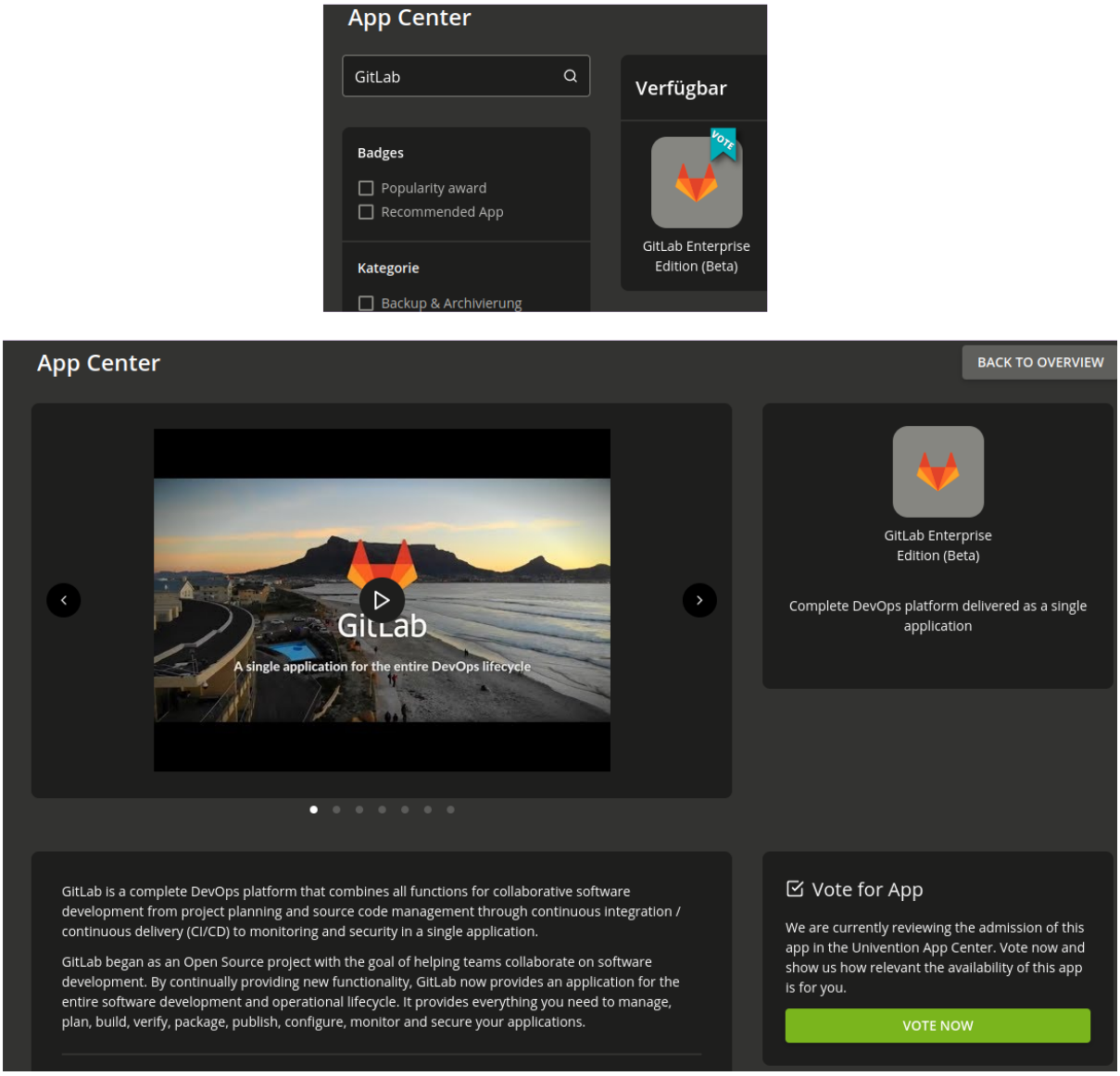


Fig. 5.3: Example *Vote Apps* in App Center overview and detail view

extensions or new modules for the UCS management system). These packages are automatically installed on the Primary Directory Node. If this is not possible, the installation is aborted. In addition, the packages are set up on all accessible Backup Directory Node systems. If several UCS systems are available in the domain, it can be selected on which system the application is to be installed.

Some applications use the container technology **Docker**. In these cases, the application (and its direct environment) is encapsulated from the rest and both security as well as the compatibility with other applications are increased.

From a technical perspective, the app is started as Docker container and joined into the UCS domain as Managed Node. A corresponding computer object is created for the Managed Node in the LDAP directory.

On the network side, the container can only be reached from the computer on which the app is installed. The app can, however, open certain ports, which can be forwarded from the actual computer to the container. UCS' firewall is correspondingly configured automatically to allow access to these ports.

If a command line is required in the app's environment, the first step is to switch to the container. This can be done by running the following command (using the fictitious app **demo-docker-app** as an example in this case):

```
$ univention-app shell demo-docker-app
```

Docker apps can be further configured via the UMC module. The app can be started and stopped and the *autostart* option be set:

Started automatically
: ensures that the app is started automatically when the server is started up.

Started manually
: prevents the app from starting automatically, but it can be started via the UMC module.

Starting is prevented
: prevents the app from starting at any time; it cannot even be started via the UMC module.

In addition, apps can also be adjusted using additional parameters. The menu for doing so can be opened using the *App Settings* button of an installed app.

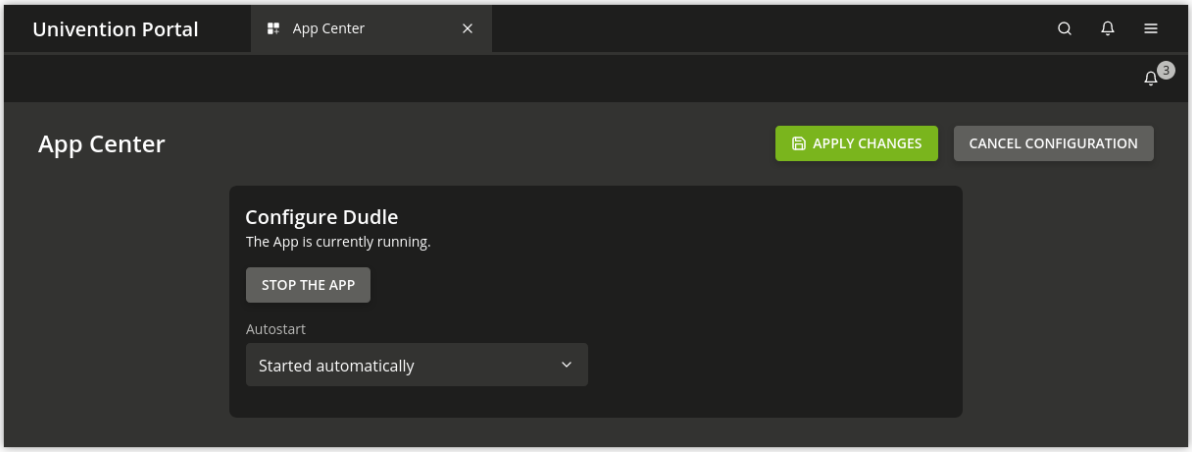


Fig. 5.4: Setting of an application in the App Center

After its installation, one or several new options are shown when clicking on the icon of an application:

Uninstall
: removes an application.

Open
: refers you to a website or a UMC module with which you can further configure or use the installed application. This option is not displayed for applications which do not have a web interface or a UMC module.

Updates for applications are published independently of the Univention Corporate Server release cycles. If a new version of an application is available, the *Upgrade* menu item is shown, which starts the installation of the new version. If updates are available, a corresponding message is also shown in the UMC module *Software update*.

Installations and the removal of packages are documented in the `/var/log/univention/management-console-module-appcenter.log` log file.

5.3 Updates of UCS systems

There are two ways to update UCS systems; either on individual systems (via UMC module *Software update* or command line) or via a computer policy for larger groups of UCS systems.

5.3.1 Update strategy in environments with more than one UCS system

In environments with more than one UCS system, the update order of the UCS systems must be borne in mind.

The authoritative version of the LDAP directory service is maintained on the Primary Directory Node and replicated on all the remaining LDAP servers of the UCS domain. As changes to the LDAP schemes (see *LDAP schemas* (page 36)) can occur during release updates, the Primary Directory Node **must always be the first system** to be updated during a release update.

It is generally advisable to update all UCS systems in one maintenance window whenever possible. If this is not possible, all not-updated UCS systems should only be one release version older compared with the Primary Directory Node.

5.3.2 Updating individual systems via Univention Management Console module

The UMC module *Software update* allows the installation of release updates and errata updates.

Fig. 5.5 shows the overview page of the module. The currently installed version is displayed under *Release updates*.

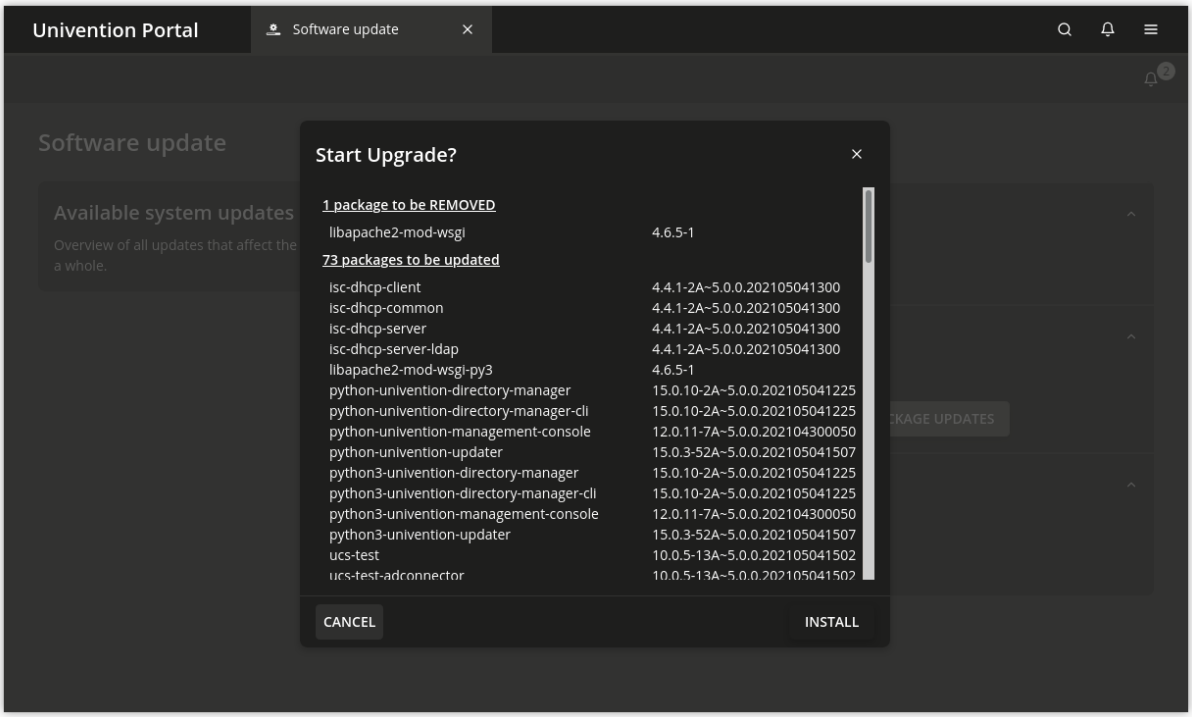


Fig. 5.5: Updating a UCS system via UMC module *Software update*

If a newer UCS version is available, a selection list is displayed. After clicking on *Install release updates* and confirmation all updates up to the respective version are installed. Before the installation process is started, a message will

be displayed informing the user of possible restrictions of the server's services during the update. Any intermediate versions are also installed automatically.

Clicking on *Install available errata updates* installs all the available errata updates for the current release and all installed components.

Check for package updates activates an update of the package sources currently entered. This can be used, for example, if an updated version is provided for a component.

The messages created during the update are written to the file `/var/log/univention/updater.log`

5.3.3 Updating individual systems via the command line

The following steps must be performed with `root` user rights.

An individual UCS system can be updated using the **`univention-upgrade`** command in the command line. A check is performed to establish whether new release or application updates are available and these are then installed if a prompt is confirmed. In addition, package updates are also performed (e.g., in the scope of an errata update).

Remote updating over SSH is not advisable as this may result in the update procedure being aborted. If updates should occur over a network connection nevertheless, it must be verified that the update continues despite disconnection from the network. This can be done, for example, using the tools **`screen`** and **`at`**, which are installed on all system roles.

The messages created during the update are written to the file `/var/log/univention/updater.log`

5.3.4 Updating systems via a policy

An update for more than one computer can be configured with an *Automatic updates* policy in the UMC modules *Computers* and *LDAP directory* (see *Policies* (page 74)).

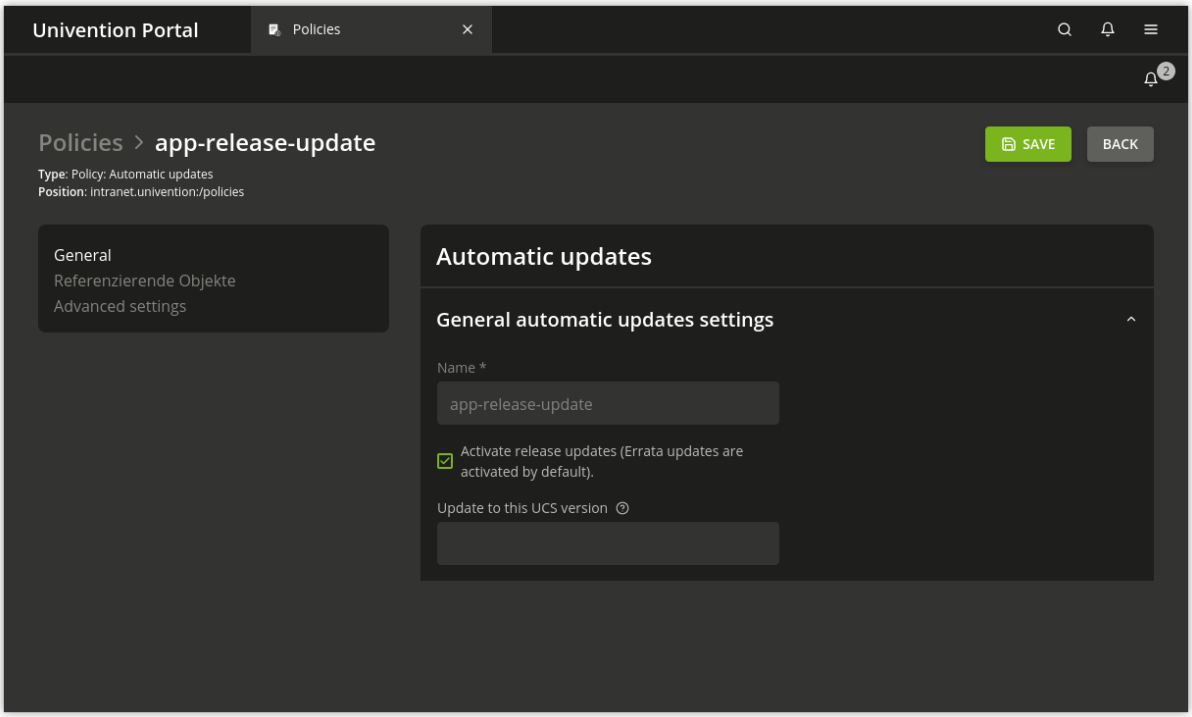


Fig. 5.6: Updating UCS systems using an update policy

A release update is only run when the *Activate release updates* selection field is activated.

The *Update to this UCS version* input field includes the version number up to which the system should be updated, for example `5.0-0`. If no entry is made, the system continues updating to the highest available version number.

The point at which the update should be performed is configured via a *Maintenance* policy (see *Specification of an update point using the package maintenance policy* (page 102)).

The messages created during the update are written to the file `/var/log/univention/updater.log`.

5.3.5 Post-processing of release updates

Once a release update has been performed successfully, a check should be made for whether new or updated join scripts need to be run.

Either the UMC module *Domain join* or the command line program **univention-run-join-scripts** is used for checking and starting the join scripts (see *How UCS systems join domains* (page 29)).

5.3.6 Troubleshooting in case of update problems

The messages generated during updates are written to the `/var/log/univention/updater.log` file, which can be used for more in-depth error analysis.

The status of the Univention Configuration Registry variables before the release update is saved in the `/var/univention-backup/update-to-TARGETRELEASEVERSION/` directory. This can then be used to check whether and which variables have been changed during the update.

5.4 Configuration of the repository server for updates and package installations

Package installations and updates can either be performed from the Univention update server or from a locally maintained repository. A local repository is practical if there are a lot of UCS systems to update as the updates only need to be downloaded once in this case. As repositories can also be updated offline, a local repository also allows the updating of UCS environments without internet access.

A local repository can require a lot of disk space.

Using the registered settings, APT package sources are automatically generated in the `/etc/apt/sources.list.d/` directory for release and errata updates as well as add-on components. If further repositories are required on a system, these can be entered in the `/etc/apt/sources.list` file.

By default the Univention repository `updates.software-univention.de` is used for a new installation.

The Univention repository contains all packages provided by Univention and Debian. A distinction is made between maintained and unmaintained packages.

- All packages in the standard package scope are in *maintained* status. Security updates are provided in a timely manner only for *maintained* packages. The list of *maintained* packages can be viewed on a UCS system in `univention-errata-level/maintained-packages.txt`.
- *unmaintained* packages are not covered by security updates or other maintenance. To check if *unmaintained* packages are installed, the command **univention-list-installed-unmaintained-packages** can be executed.

For additional repositories the installation of *unmaintained* packages is not possible by default. To enable installation, the Univention Configuration Registry Variable *repository/online/component/.*/unmaintained* (page 303) must be set to `yes`.

5.4.1 Configuration via Univention Management Console module

The *Repository server* can be specified in the UMC module *Repository Settings*.

5.4.2 Configuration via Univention Configuration Registry

The repository server to be used can be entered in the Univention Configuration Registry Variable *repository/online/server* (page 303) and is preset to `updates.software-univention.de` for a new installation.

5.4.3 Policy-based configuration of the repository server

The repository server to be used can also be specified using the *Repository server* policy in the Univention Management Console module *Computers*. Only UCS server systems for which a DNS entry has been configured are shown in the selection field (see *Policies* (page 74)).

5.4.4 Creating and updating a local repository

Package installations and updates can either be performed from the Univention update server or from a locally maintained repository. A local repository is practical if there are a lot of UCS systems to update as the updates only need to be downloaded once in this case. As repositories can also be updated offline, a local repository also allows the updating of UCS environments without internet access.

The local repository can be activated/deactivated using the Univention Configuration Registry Variable *local/repository* (page 297).

There is also the possibility of synchronizing local repositories, which means, for example, a main repository is maintained at the company headquarters and then synchronized to local repositories at the individual locations.

To set up a repository, the **univention-repository-create** command must be run as the `root` user.

The packages in the repository can be updated using the **univention-repository-update** tool. With **univention-repository-update net** the repository is synchronized with another specified repository server. This is defined in the Univention Configuration Registry Variable *repository/mirror/server* (page 303) and typically points to `updates.software-univention.de`.

An overview of the possible options is displayed with the following command:

```
$ univention-repository-update -h
```

The repository is stored in the `/var/lib/univention-repository/mirror/` directory.

5.5 Installation of further software

The initial selection of the software components of a UCS system is performed within the scope of the installation. The software components are selected relative to the functions, whereby e.g. the *Proxy server* component is selected, which then procures the actual software packages via a meta package. The administrator does not need to know the actual package names. However, individual packages can also be specifically installed and removed for further tasks. When installing a package, it is sometimes necessary to install additional packages, which are required for the proper functioning of the package. These are called package dependencies. All software components are loaded from a repository (see *Configuration of the repository server for updates and package installations* (page 98)).

Software which is not available in the Debian package format should be installed into the `/opt/` or `/usr/local/` directories. These directories are not used for installing UCS packages, thus a clean separation between UCS packages and other software is ensured.

There are several possibilities for installing further packages subsequently on an installed system, as the following sections describe.

5.5.1 Installation/deinstallation of UCS components in the Univention App Center

All software components offered in the Univention Installer can also be installed and removed at a later point in time via the Univention App Center. This is done by selecting the *UCS components* package category. Further information on the Univention App Center can be found in *Univention App Center* (page 92).

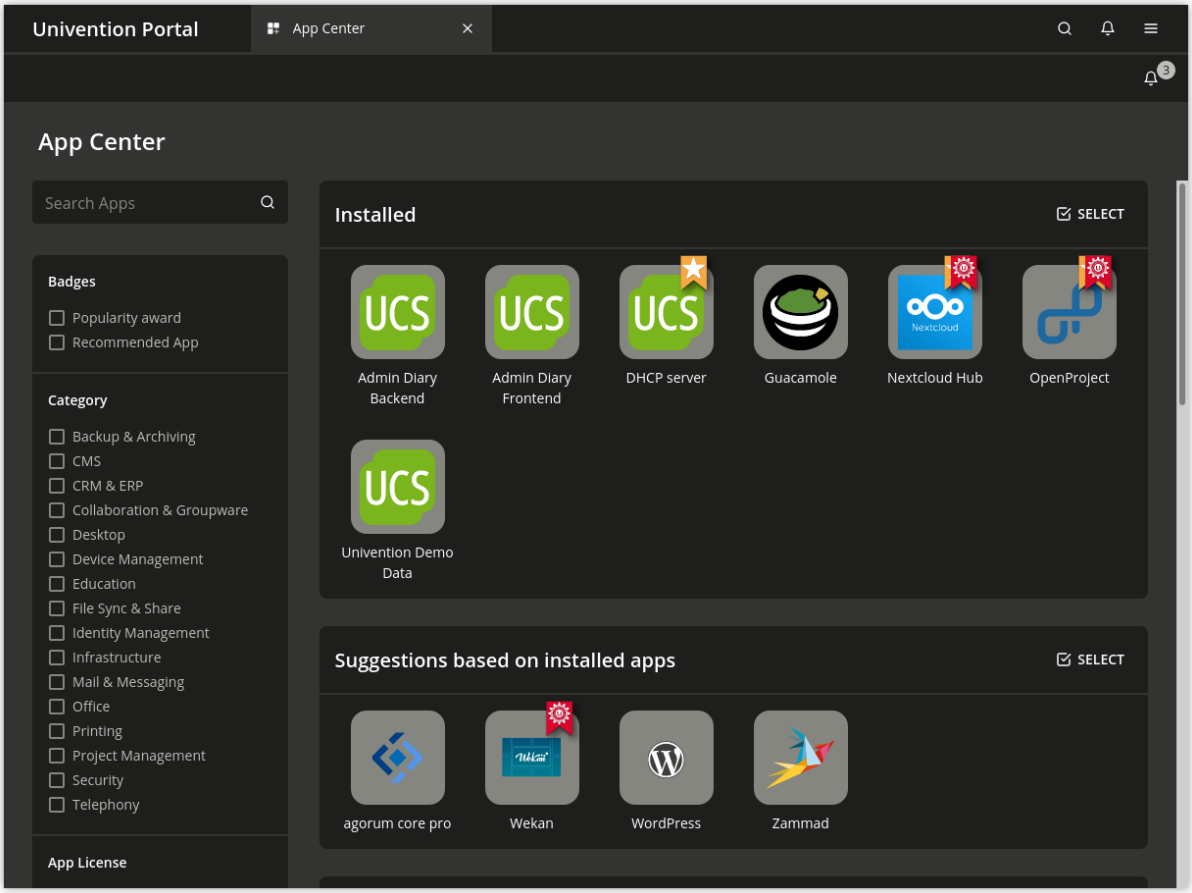


Fig. 5.7: Selection of UCS components in the App Center

5.5.2 Installation/removal of individual packages via Univention Management Console module

The UMC module *Package Management* can be used to install and uninstall individual software packages.

A search mask is displayed on the start page in which the user can select the package category or a search filter (name or description). The results are displayed in a table with the following columns:

- Package name
- Package description
- Installation status

Clicking an entry in the result list opens a detailed information page with a comprehensive description of the package.

In addition, one or more buttons will be displayed. They have the following meanings:

Install
: is displayed if the software package is not installed yet.

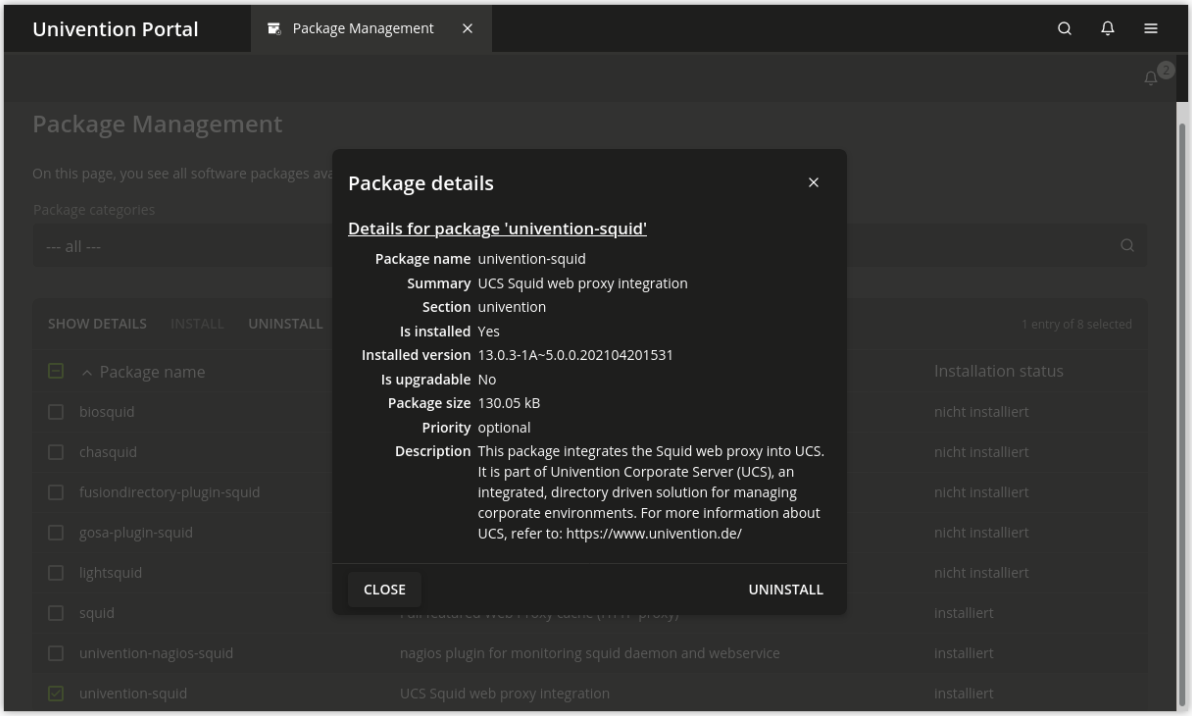


Fig. 5.8: Installing the package **univention-squid** via Univention Management Console module 'Package management'

Uninstall
is displayed if the software package is installed.

Upgrade
is displayed if the software package is installed but not updated.

Close
can be used for returning to the previous search request.

5.5.3 Installation/removal of individual packages in the command line

The following steps must be performed with `root` user rights.

Individual packages are installed using the command:

```
$ univention-install PACKAGENAME
```

Packages can be removed with the following command:

```
$ univention-remove PACKAGENAME
```

If the name of a package is unknown, the command **apt-cache search** can be used to search for the package. Parts of the name or words which appear in the description of the package are listed, for example:

```
$ apt-cache search fax
```

5.5.4 Hook scripts for administrators

Custom scripts can be called after each app installation, -upgrade or -removal. Such scripts can be used to automate repeating administrative tasks.

To use this feature custom scripts can be placed in one of the directories listed below. If such a directory does not yet exist, it can be manually created:

- `/var/lib/univention-appcenter/apps/{appid}/local/hooks/post-install.d/`
- `/var/lib/univention-appcenter/apps/{appid}/local/hooks/post-upgrade.d/`
- `/var/lib/univention-appcenter/apps/{appid}/local/hooks/post-remove.d/`

Where `{appid}` is the name of the app for which the scripts should be executed.

Script file names are only allowed to consist of lower case letters and numbers (`^[a-z0-9]+$`). Additionally scripts have to be marked as executable (**`chmod +x [filename]`**), because they are internally called by **`run-parts`**. As a consequence **`run-parts --test [directory]`** can be used to verify if and which files would be executed. Further information can be found in the manual with **`man run-parts`**.

The `/var/log/univention/appcenter.log` contains possible scripting error messages and further hints.

5.5.5 Policy-based installation/deinstallation of individual packages via package lists

Package lists can be used to install and remove software using policies. This allows central software deployment for a large number of computer systems.

Each system role has its own package policy type.

Package policies are managed in the UMC module *Policies* with the *Policy: Packages + system role*.

Table 5.1: *General* tab

Attribute	Description
Name	An unambiguous name for this package list, e.g., *mail server*.
Package installation list	A list of packages to be installed.
Package removal list	A list of packages to be removed.

The software packages defined in a package list are installed/uninstalled at the time defined in the *Maintenance* policy (for the configuration see *Specification of an update point using the package maintenance policy* (page 102)).

The software assignable in the package policies are also registered in the LDAP.

5.6 Specification of an update point using the package maintenance policy

A *Maintenance* policy (see *Policies* (page 74)) in the UMC modules for computer and domain management can be used to specify a point at which the following steps should be performed:

- Check for available release updates to be installed (see *Updating systems via a policy* (page 97)) and, if applicable, installation.
- Installation/deinstallation of package lists (see *Policy-based installation/deinstallation of individual packages via package lists* (page 102))
- Installation of available errata updates

Alternatively, the updates can also be performed when the system is booting or shut down.

Table 5.2: *General* tab

Attribute	Description
Perform maintenance after system startup	If this option is activated, the update steps are performed when the computer is started up.
Perform maintenance before system shutdown	If this option is activated, the update steps are performed when the computer is shut down.
Use Cron settings	If this flag is activated, the fields *Month*, *Day of week*, *Day*, *Hour* and *Minute* can be used to specify an exact time when the update steps should be performed.
Reboot after maintenance	This option allows you to perform an automatic system restart after release updates either directly or after a specified time period of hours.

5.7 Central monitoring of software installation statuses with the software monitor

The software monitor is a database in which information is stored concerning the software packages installed across all UCS systems. This database offers an administrator an overview of which release and package versions are installed in the domain and offers information for the step-by-step updating of a UCS domain and for use in identifying problems.

The software monitor can be installed from the Univention App Center with the application **Software installation monitor**. Alternatively, the software package **univention-pkgdb** can be installed. Additional information can be found in *Installation of further software* (page 99).

UCS systems update their entries automatically when software is installed, uninstalled or updated. The system on which the software monitor is operated is located by the DNS service record `_pkgdb._tcp`.

The software monitor brings its own UMC module *Software monitor*. The following functions are available:

Systems
: allows to search for the version numbers of installed systems. It is possible to search for system names, UCS versions and system roles.

Packages
: allows to search in the installation data tracked by the package status database. Besides searching for a *Package name* there are various search possibilities available for the installation status of packages:

  Selection state
  : The *selection state* influences the action taken when updating a package. `Install` is used to select a package for installation. If a package is configured to `Hold` it will be excluded from further updates. There are two possibilities for uninstalling a package: A package removed with `DeInstall` keeps locally created configuration data, whilst a package removed with `Purge` is completely deleted.

  Installation state
  : The *installation state* describes the status of an installed package in relation to upcoming updates. The normal status is `Ok`, which leads to a package being updated when a newer version exists. If a package is configured to `Hold` it will be excluded from the update.

  Package state
  : The *package state* describes the status of a setup package. The normal status here is `Installed` for installed packages and `ConfigFiles` for removed packages. All other statuses appear when the package's installation was canceled in different phases.

If you do not wish UCS systems to store installation processes in the software monitor (e.g., when there is no network connection to the database), this can be arranged by setting the Univention Configuration Registry Variable *pkgdb/scan* (page 302) to `no`.

Should storing be reactivated at a later date, the command **univention-pkgdb-scan** must be executed to ensure that package versions installed in the meanwhile are also adopted in the database.

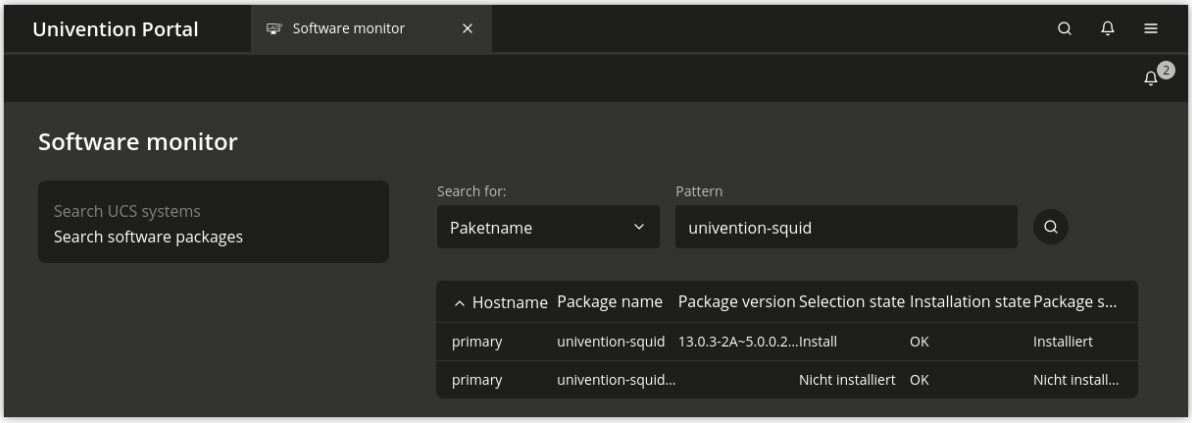


Fig. 5.9: Searching for packages in the software monitor

The following command can be used to remove a system's program inventory from the database again:

```
$ univention-pkgdb-scan --remove-system [HOSTNAME]
```

CHAPTER

SIX

USER MANAGEMENT

UCS integrates central identity management. All user information are managed centrally in UCS via the Univention Management Console module *Users* and stored in the LDAP directory service.

Note: The user management is part of Univention Nubus in the *Directory Manager* component. For more information about Nubus, refer to *What is Univention Nubus?* (page 2)

All the services integrated in the domain access the central account information, i.e., the same username and password are used for the user login to a Windows client as for the login on the IMAP server.

The domain-wide management of user data reduces the administrative efforts as changes do not need to be subsequently configured on different individual systems. Moreover, this also avoids subsequent errors arising from inconsistencies between the individual datasets.

User account types

There are three different types of user accounts in UCS:

1. **Normal user accounts** have all available properties. These users can log in to UCS or Windows systems and, depending on the configuration, also to the installed Apps. The users can be administered via the UMC module *Users* (see *User management through Univention Management Console module* (page 106)).
2. **Address book entries** can be used to maintain internal or external contact information. These contacts can't sign in to UCS or Windows systems. Address book entries can be managed via the UMC module *Contacts*.
3. **Simple authentication account**: With a simple authentication account, a user object is created, which has only a username and a password. With this account, only authentication against the LDAP directory service is possible, but no login to UCS or Windows systems. Simple authentication accounts can be accessed via the UMC module *LDAP directory* (see *LDAP directory browser* (page 73)).

Recommendation for username definition

One very important and required attribute for user accounts is the *username*. To avoid conflicts with the different tools handling user accounts in UCS, adhere to the following recommendations for the definition of usernames:

- Only use lower case letters (`a-z`), digits (`0-9`) and the hyphen (`-`) from the ASCII character set for usernames.
- The username starts with a lower case letter from the ASCII character set. The hyphen is not allowed as last character.
- In UCS the username has at least a length of 4 characters and at most 20 characters.

The recommendation results in the following regular expression: `^[a-z][a-z0-9-]{2,18}[a-z0-9]$`.

Besides the recommendation, usernames also contain underscores (_) and upper case ASCII letters in practice. Consider the recommendation as a guideline and not a rule and keep potential side-effects in mind when defining usernames outside the recommendation.

6.1 User management through Univention Management Console module

This section describes the user management through the UMC module *Users*.

6.1.1 User creation wizard

To create users, administrators can use the simplified user creation wizard, as shown in the following screenshots.

You open the wizard by clicking the *Add* button in the *Users* module. In the first page *Add a new user*, you select the *Container* where you want to locate the user object and if you want to create a user account using a template.

When you click *Next*, you see the page in Fig. 6.1.

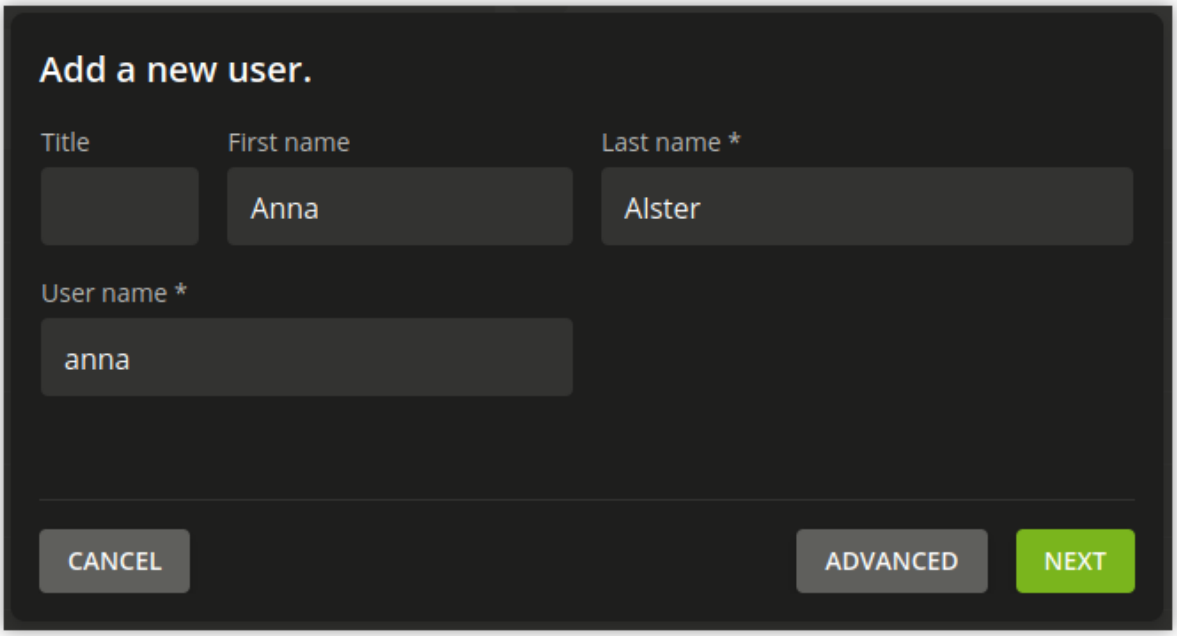


Fig. 6.1: Creating a user in the UMC module *Users*

With *Next*, the *Users* module shows the third page as in Fig. 6.2, where you can define the initial password.

As an alternative the user may set the initial password themselves, if the UCS domain has the **Self Service** app installed. To allow the user setting the initial password by themselves, you must define their external email address.

To define an external email address, open the *Advanced* user account settings, click the tab *Contact* and provide an external email address in the field *E-mail address*. The self service app sends an email to the user's external email address with a link and a token. The user can use the link to set their initial password and unlock the user account. For more information, refer to *Password management via Self Service app* (page 119).

By default, the *Users* module shows a simplified wizard for creating a user. The wizard only requests the most important settings. To see all user account attributes as in Fig. 6.4, click *Advanced* in the wizard.

You can deactivate the simplified wizard by setting the Univention Configuration Registry Variable *directory/manager/web/modules/users/user/wizard/disabled* (page 293) to `true` and restarting the **univention-management-console-server**.

If you want to define the user's primary email address in the simplified wizard, you can activate the field by setting the Univention Configuration Registry Variable `directory/manager/web/modules/users/user/properties/mailPrimaryAddress/required` to `true` and restarting the **univention-management-console-server**. The wizard's second page then asks for the user's primary email address, as shown in Fig. 6.5.

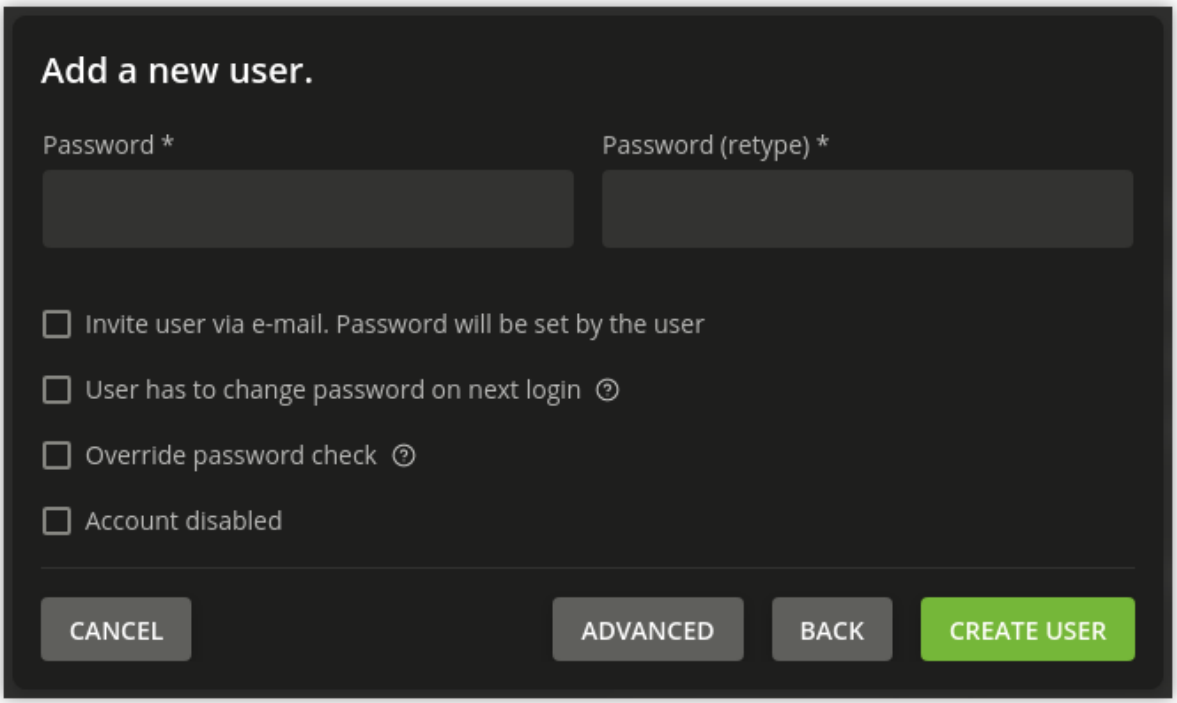


Fig. 6.2: Password setting for a user

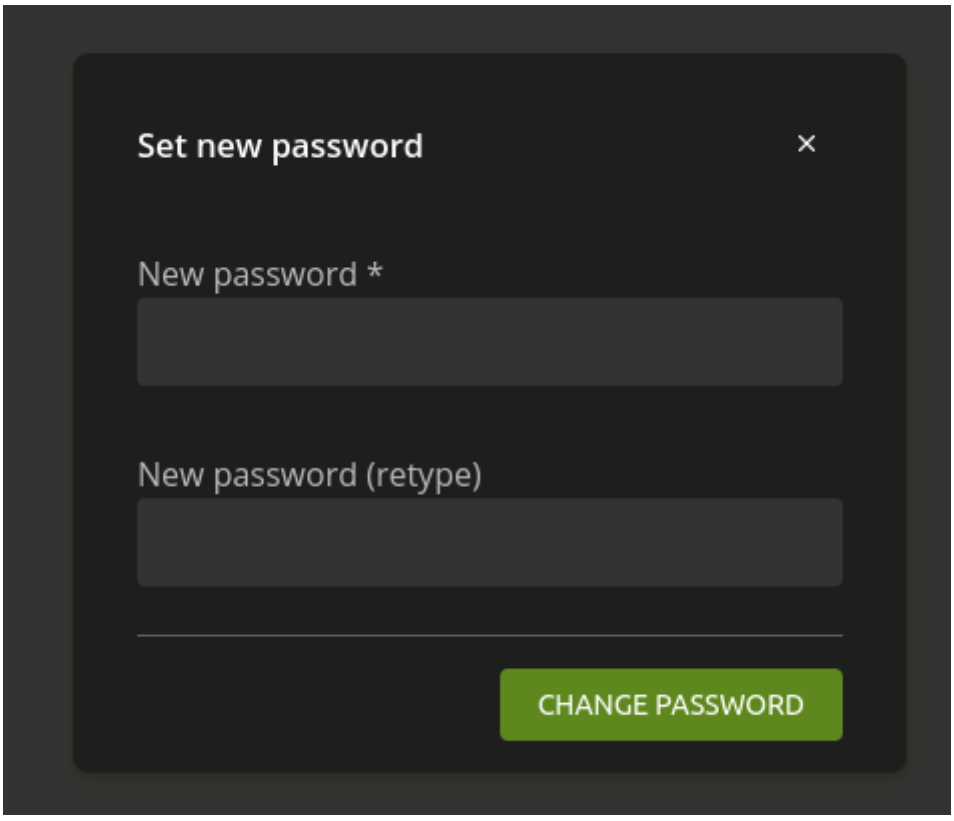


Fig. 6.3: Initial user password

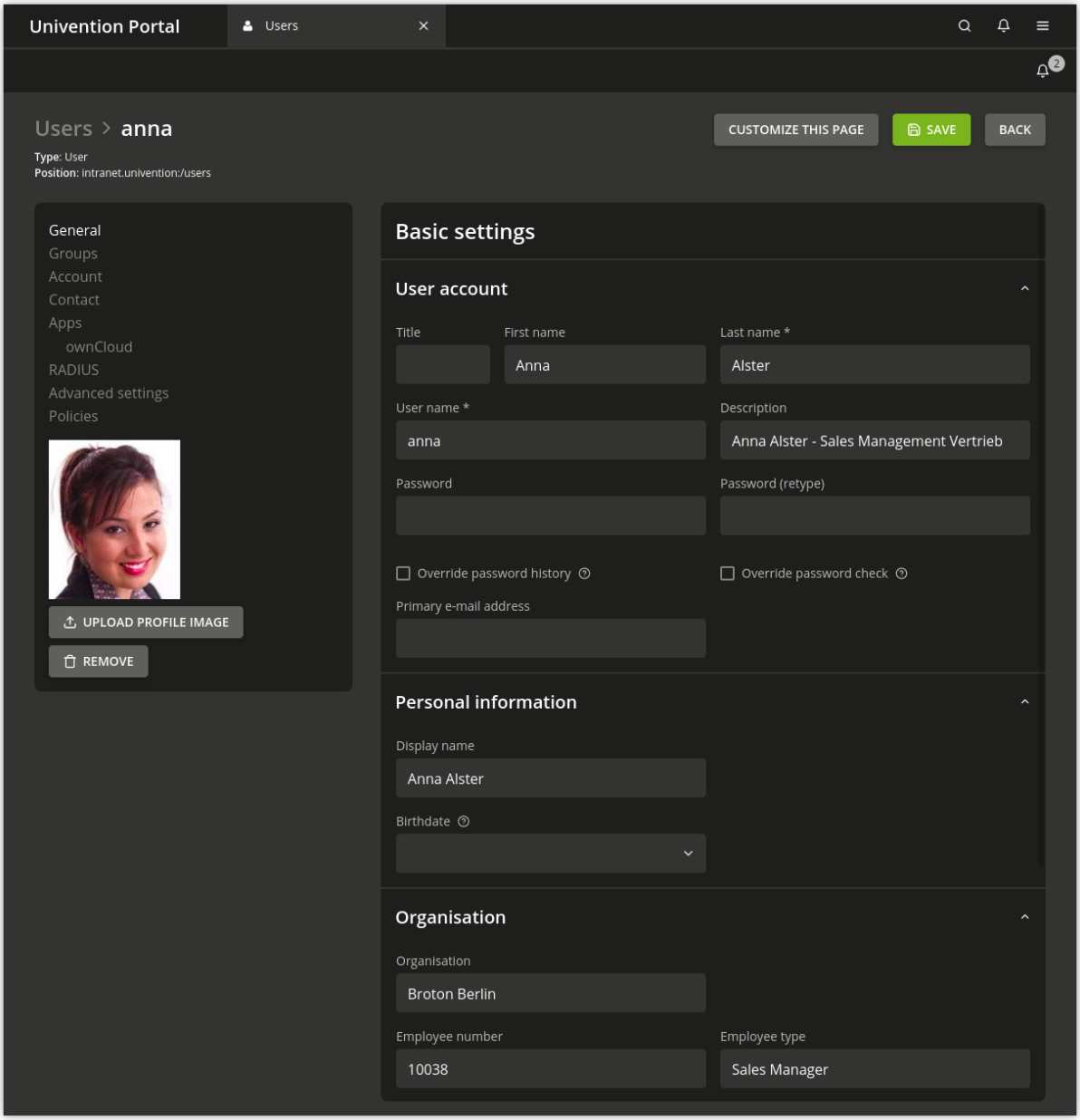


Fig. 6.4: Advanced user settings

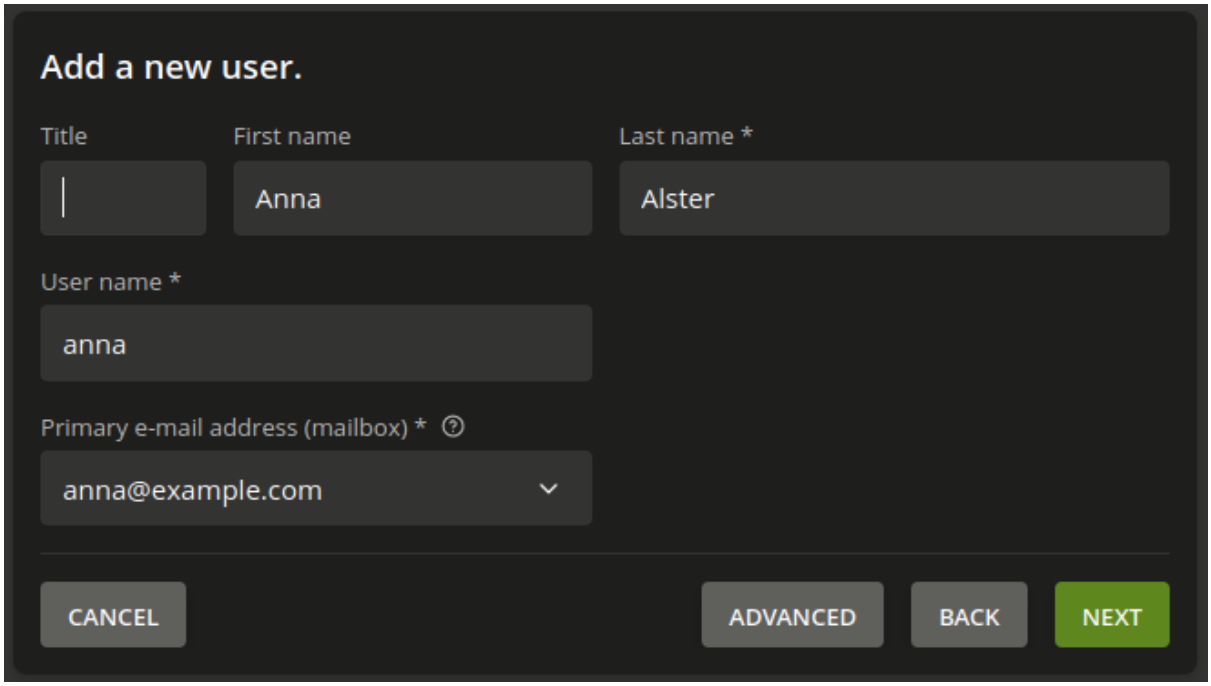


Fig. 6.5: Require setting the user's primary email address in the wizard

6.1.2 User management module - General tab

Table 6.1: *General* tab

Attribute	Description
Title	The title of the user is to be entered here.
First name	The first name of the user is to be entered here.
Last name	The last name of the user is to be entered here.
Username	This is the name, by which the user logs into the system. For recommended characters for the user, see *Recommendation for username definition* (page 105). In order to ensure compatibility to non-UCS systems the creation of users which are only distinguished from each other by upper and lower case letters is prevented. Thus, if the username `smith` already exists, then the username `Smith` cannot be created. By default it is not possible to create a user with the same name as an existing group. If the Univention Configuration Registry Variable *`directory/manager/user_group/uniqueness`* (page 292) is set to `false`, this check is removed.
Description	Arbitrary descriptions for the user can be entered here.
Password	The user's password has to be entered here.
Password (retype)	In order to avoid spelling errors, the user's password has to be entered for a second time.
Override password history	By checking this box, the password history is overridden for this user and for this password change. This means, with this change the user can be assigned a password which is already in use. Further details on user password management can be found in *User password management* (page 115).
Override password check	By checking this box, the requirements for the length of the password and for password quality checks are overridden for this user and for this password change. This means, the user can e.g. be assigned a shorter password than would be possible according to the defined minimum length. Further details on the password policies for users can be found in *User password management* (page 115).
Primary email address (mailbox)	The email address of the user is declared here, see *Assignment of email addresses to users* (page 262).
Display name	The display name is automatically composed of the first and surnames. It generally does not need to be changed. The screen name is used for the synchronization with Active Directory and Samba/AD among other things.
Birthday	This field is used to save a user's birthday.
Organization	The organization is to be entered here.
Employee number	Numbers for staff members can be entered in this field.
Employee type	The category of the staff member can be entered here.
Superior	The superior of the user can be selected here.
Picture of the user (JPEG format)	This mask can be used to save a picture of the user in LDAP in JPEG format. In the default settings the file size is limited to 512 kilobytes.

6.1.3 User management module - Groups tab

Table 6.2: *Groups* tab

Attribute	Description
Primary group	This selection list can be used for specifying the user's primary group. All the groups registered in the domain are open for selection. By default, the group `Domain Users` is preset.
Groups	Here it is possible to set further group memberships for the user in addition to the primary group.

6.1.4 User management module - Account tab

Table 6.3: *Account* tab

Attribute	Description
Account is deactivated	The *Account is deactivated* checkbox can be used to deactivate the user account. If the checkbox is selected, the user cannot log into the system. This affects all authentication methods. This is typically used when a user leaves the company. In a heterogeneous environment, an account deactivation might also be caused by external tools.
Account expiry date	A date is specified in this input field on which the account will automatically be locked. This is practical for user accounts that only need to be active for a certain period of time, e.g., for interns. If the date is deleted or replaced by a different, future date, the user will regain the right to sign in.
User has to change password on next login	If this checkbox is ticked, then the user has to change their password during the next login procedure.
Password expiry date	If the password is subject to an expiry date, then this date is displayed in this entry field. This entry field cannot be edited directly, see *User password management* (page 115). If a password expiry interval is defined, the password expiry date is automatically adjusted when passwords are changed. If no *Expiry interval* is declared, the old expiry date will be deleted and no new date will be set.
Unlock lockout	If the account has automatically been locked temporarily for security reasons, usually because the user has entered the password incorrectly too often, this checkbox can be used to unlock the account again manually before the lockout is lifted automatically when the lockout duration has passed. This temporary account lockout can happen if a corresponding domain wide policy setting has been defined by an administrator. There are three different mechanisms that may trigger lockout if configured properly: • Failed PAM authentication attempts to an UCS server (see *Automatic lockout of users after failed login attempts* (page 127)). • Failed LDAP authentication attempts (if the **ppolicy** overlay has been activated and configured). • Failed Samba/AD authentication attempts (if the Samba domain `passwordsettings` have been configured).
Lockout till	If the account has automatically been locked temporarily for security reasons, usually because the user has entered the password incorrectly too often, this field shows the time when the account automatically gets unlocked.
Activation date	If a user account shall only become usable at a later date, this can be set here. A cron job periodically checks if accounts need to be activated. It runs every 15 minutes by default. When saving the changes, the account is automatically marked as deactivated in case a date in the future has been specified.
Windows home drive	If the Windows home directory for this user is to show up on a different Windows drive than that specified by the Samba configuration, then the corresponding drive letter can be entered here, e.g. `M:`.
Windows home path	The path of the directory which is to be the user's Windows home directory, is to be entered here, e.g. `\ucs-file-serversmith`.
Windows logon script	The user-specific logon script relative to the NETLOGON share is entered here, e.g. `user.bat`.
Windows profile directory	The profile directory for the user can be entered here, e.g. `\ucs-file-serveruserprofile`.
Relative ID	The relative ID (RID) is the local part of the SID. If a user is to be assigned a certain RID, the ID in question can be entered in this field. If no RID is assigned, the next available RID will automatically be used. The RID cannot be subsequently changed. Integers from 1000 upwards are permitted. RIDs below 1000 are reserved to standard groups and other special objects.
Samba privilege(s)	This selection mask can be used to assign a user selected Windows systems rights, for example the permission to join a system to the domain.
Permitted times for Windows logins	This input field contains time periods for which this user can sign in to Windows computers. If no entry is made in this field, the user can sign in at any time of day.

6.1.5 User management module - Contact tab

Table 6.4: *Contact* tab

Attribute	Description
Email address(es)	Additional email addresses can be saved here. These are not evaluated by the mail server. The values of this attribute are stored in the LDAP attribute `mail`. Most address book applications using an LDAP search function will search for an email address by this attribute.
Telephone number(s)	This field contains the user's business phone number.
Room number(s)	The room number of the user.
Department number(s)	The department number of the user can be entered here.
Street	The street and house number of the user's business address can be entered here.
Postal code	This field contains the postal code of the user's business address.
City	This field contains the city of the user's business address.
Private telephone number(s)	The private fixed network phone number can be entered here.
Mobile telephone number(s)	The user's mobile numbers can be entered here.
Pager telephone number(s)	Pager numbers can be entered here.
Private postal address(es)	One or more of the user's private postal addresses can be entered in this field.

6.1.6 User management module - Mail tab

This tab is displayed in the advanced settings.

The settings are described in *Assignment of email addresses to users* (page 262).

6.1.7 User management module - Options tab

Table 6.5: *(Options)* tab

Attribute	Description
Public key infrastructure account	If this checkbox is not ticked, the user will not be assigned the object class `pkiUser`.

6.2 User activation for apps

Many apps from the App Center are compatible with the central identity management in UCS. This allows system administrators to activate the users for apps. In some cases, app specific settings for the user can be made. This depends on the app and how it uses the identity management.

Once an app with user activation is installed in the UCS environment, it will appear with the logo in the *Apps* tab of the user in the UMC module *Users*. With a tick in the checkbox the user is activated for the app. If the app offers specific settings another tab with the name of the app will appear to set these parameters. The app activation and the parameters are stored at the user object in the LDAP directory service.

To withdraw a user activation for an app, it is sufficient to deselect the checkbox.

When the app is uninstalled, the checkbox of the user activation for the app is removed from the *Apps* tab of the user in the UMC module.

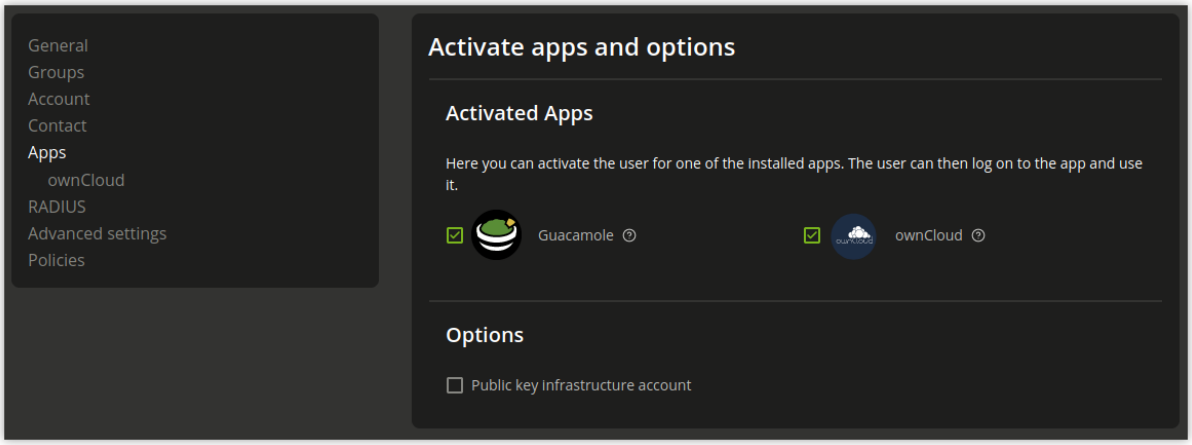


Fig. 6.6: User activation for installed apps

6.3 User password management

Most internet users find it difficult to choose the right password. The password is the key to accessing user accounts, even in a UCS domain. Passwords that are difficult to guess and regular passwords changes are an essential element of the system security of a UCS domain. To prevent users from choosing passwords that are too easy, administrators can configure several properties in a *password policy*.

This section describes how to define password policies, such as a minimum password length and an expiration time interval. UCS applies the password policy when users change their passwords.

UCS stores the user password for every user as hash in different attributes of the corresponding LDAP user account object:

`krb5Key`
: stores the Kerberos password.

`userPassword`
: stores the Unix password. Other Linux distributions store it in `/etc/shadow`.

`sambaNTPassword`
: stores the NT password hash used by Samba.

See also:

Creating Secure Passwords[25] by German Federal Office of Information Security
: for more information and tips about creating a secure and good password.

6.3.1 Password policy types

UCS has several types of password policy settings as outlined in this section. What policy applies depends on who runs the password change and if the UCS domain has Samba installed through the app **`Active Directory Domain Controller`**.

Password Policy in UDM
: The *Password Policy* is a UDM policy that applies to user password changes done through UMC modules which in turn use UDM in the backend. The *Password Policy* applies, when an **administrator** changes a user's password through UMC or UDM. It also applies, when a **user** changes their password and if the UCS domain **hasn't** Samba installed.

  UCS defines a default password policy. *Password policy settings in UMC* (page 117) describes the available settings for the *Password Policy*.

[25] https://www.bsi.bund.de/EN/Themen/Verbraucherinnen-und-Verbraucher/Informationen-und-Empfehlungen/Cyber-Sicherheitsempfehlungen/Accountschutz/Sichere-Passwoerter-erstellen/sichere-passwoerter-erstellen_node.html

You can enhance the *Password quality check* with Univention Configuration Registry Variables mentioned in *Password policy settings in UMC* (page 117).

You can create additional password policies and assign them to user account objects in the LDAP directory tree. For more information about policies, see *Policies* (page 74).

Important: When you have Samba installed, design the password requirement settings of the user password policy identical to the Samba domain object as described in *Password settings for Windows clients when using Samba* (page 118).

Password policy for the Samba domain
When you have Samba installed in your UCS domain, the Samba domain has it's own password policy. The Samba password policy **always** applies, when a **user** changes their password, regardless of the used service, through Univention Portal, User Self Service, Microsoft Windows, or Kerberos.

To configure the password policy for the Samba domain, see *Password settings for Windows clients when using Samba* (page 118).

See also:

***Installation* (page 171) of Samba**
for more information about the installation of Samba.

***Services for Windows* (page 171)**
for general information about Samba providing Services for Windows.

6.3.2 Change the user password

Changing the user password has the following triggers:

1. The systems requires the user to change their password, for example, because the password reached the expiration interval.
2. Through a setting at the user account, an administrator requests the user to change their password upon next sign-in.
3. The user decides to change their password.

When a user decides to change their password, they can use the following ways:

Univention Portal
Every UCS domain has the *Portal* installed. To change the password, use the following steps:

1. Sign in to the *Univention Portal*.
2. Navigate to the user menu. It's the *"burger menu"* in top right corner.
3. Select *User settings* ‣ *Change your password*.
4. Provide your current password and set a new password. Retype it and confirm.

User Self Service
The *User self services* (page 119) is a dedicated app in the *Univention App Center* (page 92). It offers a direct link to the password change so that administrators can add a prominent tile to the Univention Portal for the password change. Furthermore, the app offers a way to reset the user password when they forgot it.

Microsoft Windows
Users can change their user password through their Microsoft Windows client that's joined in the UCS domain through Samba.

Kerberos
Users can change their user password through clients that have joined in the UCS domain through Kerberos. They can use the default features of those clients to change the password.

For more information about joining Ubuntu and Linux systems to a UCS domain and the integration with Kerberos, see *Extended domain services documentation* [2].

6.3.3 Password policy settings in UMC

With the password policy settings in UMC administrators can define the minimum password length, the expiry interval and the password history length. Fig. 6.7 shows the password policy settings in UMC. Following the figure, you find a reference of the available settings.

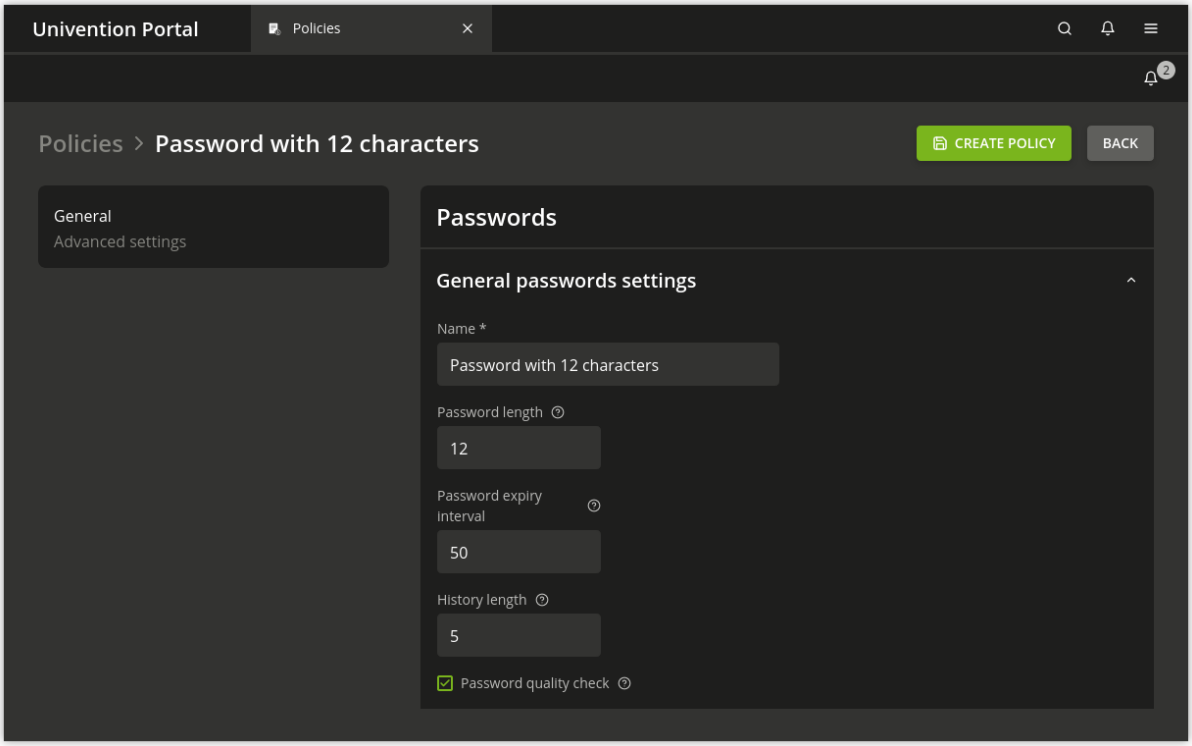


Fig. 6.7: Configuring a password policy

On the *General* tab of a password policy, you can configure the following settings.

History length
: The password history saves the last used password hashes. The *History length* determines the length of that history, for example if the history stores the last three or the last seven passwords. Users can't reuse passwords from the password history for setting a new password. UCS doesn't store the passwords retroactively.

  To deactivate the validation for the password history, set the value to `0`.

  Example
  : If UCS stored ten passwords, and you reduce the value for the password history length to `3`, UCS deletes the oldest seven passwords from the password history during the next password change. If you then change the password history length, the number of stored passwords stays at three and increases by each password change.

Password length
: The *Password length* is the minimum length in characters that a user password must comply with. If you don't set a value, UCS applies the minimum length of `8` characters.

  The default value always applies if you don't set a policy and you activated the *Override password check* checkbox. This means it even applies if you deleted the *default-settings* password policy.

  To deactivate the validation for the password length, set the value to `0`.

You can configure a default value per UCS system through the Univention Configuration Registry Variable `password/quality/length/min` (page 302). The setting applies to users that aren't subject to a *UDM password policy*.

Password expiry interval
A *Password expiry interval* demands regular password changes. UCS requires a user to change their password during sign-in to UCS web interfaces, to Kerberos, and on UCS systems when the expiry interval in days passed. UCS displays the remaining validity of the user password in the user management module at *Password expiry date* in the *Account* tab.

To deactivate the *Password expiry interval*, leave the value blank.

Password quality check
If you activate the option *Password quality check*, UCS runs additional password checks, including dictionary checks, for password changes in UMC and Kerberos.

You configure the quality checks through the following Univention Configuration Registry variables. For more information, refer to linked variable descriptions.

You can enforce the following checks:

- `password/quality/credit/digits` (page 302)
- `password/quality/credit/upper` (page 302)
- `password/quality/credit/lower` (page 302)
- `password/quality/credit/other` (page 302)
- `password/quality/forbidden/chars` (page 302)
- `password/quality/required/chars` (page 302)
- `password/quality/mspolicy` (page 302)

Important: To apply the *password quality check* on all UCS sign-in systems, you need to set the Univention Configuration Registry Variables on **all** UCS sign-in servers.

6.4 Password settings for Windows clients when using Samba

With the Samba domain object, you can set the requirements for user account passwords in a Samba domain.

You can manage the Samba domain object through the UMC module *LDAP directory*. The Samba domain object locates in the `samba` container and has the domain's NetBIOS name. You find the `samba` container under the LDAP base.

Important: It's a strong recommendation to design the password requirement settings of the Samba domain object identical to the user password policy as described in *User password management* (page 115).

In the *Password* section on the *General* tab of the *Samba Domain* object, you can configure the following settings.

Password length
The minimum number of characters for a user password.

Password history
UCS stores password changes in the form of hashes. Users can't use passwords from history when setting a password. For example, with a password history value of `5`, user must set five other passwords before they can reuse a password from the history.

Minimum password age
Defines the period of time that must elapse, before users can change their password.

Maximum password age
: Defines the maximum age for a password. When the period of time is over, UCS requires the user to change their password upon next sign-in.

  To define an infinite period of time, leave the value empty.

Password must meet complexity requirements
: Activating the checkbox activates Microsoft Password complexity requirements[26]. A tooltip shows the required characters in a password. The library `Passfilt.dll` enforces the complexity requirements. Administrators can't change them.

6.5 User self services

For information about setting a user password policy, refer to *User password management* (page 115).

6.5.1 Password change by user via UCS portal page

Every logged in user can change their own password by opening the menu via the hamburger icon in the top right corner and selecting *User settings* ‣ *Change password*. The change is performed directly via the PAM stack (see *Authentication / PAM* (page 162)) and is then available centrally for all services.

6.5.2 Password management via Self Service app

By installing the UCS components **`Self Service Backend`** and **`Self Service`** in the domain via the *App Center*, users are enabled to take care of their password management without administrator interaction.

The **`Self Service`** app creates its own portal, accessible at the web URI `/univention/selfservice/`, which bundles all its functionality. The original portal has the same entries registered at its user menu. They allow users to update their password given their old password as well as to reset their lost password by requesting a token to be sent to a previously registered contact email address. The token has to be entered on the dedicated password reset web page.

The following Univention Configuration Registry Variables can be used to activate or deactivate individual features of the password management.

`umc/self-service/passwordreset/backend/enabled`
: Activates or deactivates the backend of the *Password forgotten* page. This Univention Configuration Registry Variable has to be set on the systems that is defined as **`Self Service backend`** via the Univention Configuration Registry Variable *`self-service/backend-server`* (page 304), since requests regarding these variables are forwarded to the **`Self Service backend`**.

`umc/self-service/protect-account/backend/enabled`
: Activates or deactivates the backend of the *Protect account* page. This Univention Configuration Registry Variable has to be set on the systems that is defined as **`Self Service backend`** via the Univention Configuration Registry Variable *`self-service/backend-server`* (page 304), since requests regarding these variables are forwarded to the **`Self Service backend`**.

`umc/self-service/service-specific-passwords/backend/enabled`
: Activates or deactivates the backend for service specific passwords. Currently, only the service RADIUS is supported. Find more information in *Service specific password* (page 230).

Those variables also activate or deactivate the corresponding entries in the portal. However, you can also adjust them manually, they are in fact just normal portal entries.

[26] https://learn.microsoft.com/de-de/previous-versions/windows/it-pro/windows-10/security/threat-protection/security-policy-settings/password-must-meet-complexity-requirements

6.5.3 Contact information

Additional personal data can be stored in LDAP with the users account. This may include a picture, the users private address and other contact information. By default only administrators can modify them. As an alternative selected attributes may be unlocked for the user to change himself. The user then can do this using the **Self Service** app.

For this the following Univention Configuration Registry Variables must be configured:

self-service/ldap_attributes

This variable configures the *LDAP* attributes a user can modify at its own user account. The names of the attributes must be separated by comma. This variable must be set on Primary Directory Node (and Backup Directory Nodes).

self-service/udm_attributes

This variable configures the *UDM* attributes a user can modify. The names of the attributes must be separated by comma. This variable must be set on all hosts, where the **Self Service** app is installed (incl. Primary Directory Node).

self-service/udm_attributes/read-only

This variable sets *UDM* attributes specified in Univention Configuration Registry Variable *self-service/udm_attributes* (page 120) to read-only. Use a comma-separated list for multiple values. Set this variable on all hosts, where the **Self Service** app is installed, including Primary Directory Node.

To prevent this variable's intended behavior from being prohibited, remove the *LDAP* attributes specified in Univention Configuration Registry Variable *self-service/ldap_attributes* (page 120) that should be read-only. Otherwise, these *LDAP* attributes will keep the corresponding *UDM* attributes writable.

umc/self-service/profiledata/enabled

This variable must be set to `true` on all involved server systems to enable the mechanism.

umc/self-service/allow-authenticated-use

This variable defines whether the specification of user name and password is necessary when opening and modifying your own user profile if you are already logged in to Univention Portal.

As of UCS 4.4-7, this Univention Configuration Registry Variable is automatically set to `true` when the **Self Service** is installed for the first time. The `true` value activates the use of an existing Univention Portal session. The fields *Username* and *Password* are then automatically filled in or no longer displayed.

Systems upgraded to UCS 4.4-7 will retain the old behavior by automatically setting the value to `false`. Note that this variable must be set to the same value on all participating systems where the **Self Service** app is installed (incl. Primary Directory Node).

Both `*attributes` variables must match each other. The names of the attributes and their mapping can be fetched from the following command:

```
$ python3 -c 'from univention.admin.handlers.users.user import mapping;\
  print("\n".join( \
  map("{0[0]:>30s} {0[1][0]:<30s}".format, sorted(mapping._map.items())))) \
  )'
```

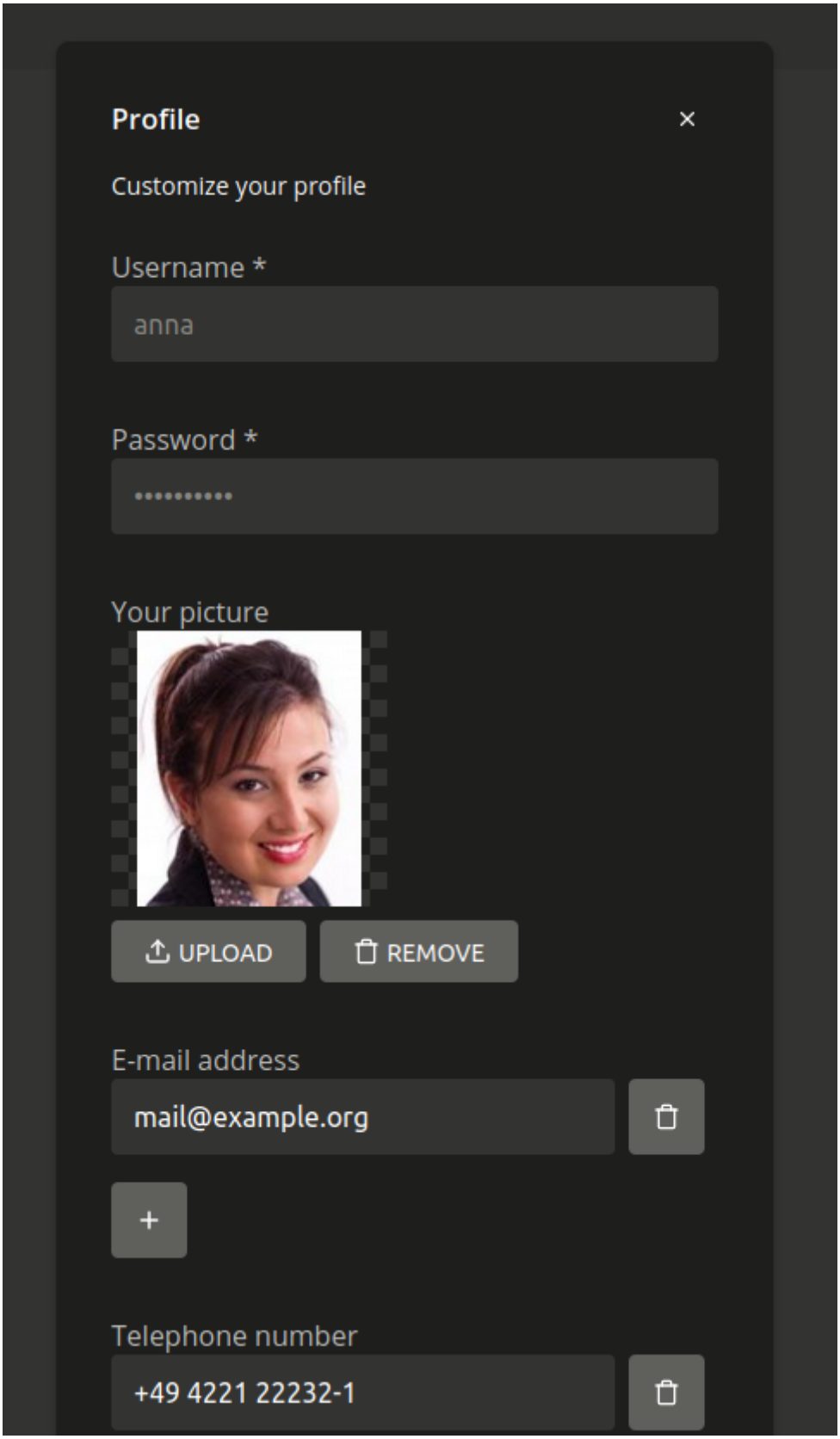


Fig. 6.8: User profile self-service

6.5.4 Self registration

The Self Service allows for users to register themselves, which will create a user account that has to be verified via email.

User accounts that are created via the Self Service will have the `RegisteredThroughSelfService` attribute of the user set to `TRUE` and the `PasswordRecoveryEmailVerified` attribute set to FALSE. After the user has verified their account via the verification email the `PasswordRecoveryEmailVerified` attribute will be set to `TRUE`.

Account creation

Aspects about the *Create an account* page and the account creation itself can be configured with the following Univention Configuration Registry Variables. These Univention Configuration Registry Variables have to be set on the systems that is defined as **`Self Service Backend`** via the Univention Configuration Registry Variable *self-service/backend-server* (page 304), since requests regarding these variables are forwarded to the Self Service backend.

umc/self-service/account-registration/backend/enabled

With this variable the account registration can be disabled/enabled.

umc/self-service/account-registration/usertemplate

With this variable a *user template* (page 129) can be specified that will be used for the creation of self registered accounts.

umc/self-service/account-registration/usercontainer

With this variable a container can be specified under which the self registered users are created.

umc/self-service/account-registration/udm_attributes

This variable configures which UDM attributes of a user account are shown on the *Create an account* page of the Self Service. The names of the UDM attributes must be provided as a comma separated list.

umc/self-service/account-registration/udm_attributes/required

This variable configures which of the UDM attributes set via the Univention Configuration Registry Variable *umc/self-service/account-registration/udm_attributes* (page 122) are required for the user to provide. The names of the UDM attributes must be provided as a comma separated list.

Verification email

After a user has clicked on *Create account*, they will see a message that an email for the account verification has been sent.

Aspects about the *verification email* and the verification token can be configured with the following Univention Configuration Registry Variables. These Univention Configuration Registry Variables have to be set on the **`Self Service Backend`** that is defined via the Univention Configuration Registry Variable *self-service/backend-server* (page 304), since requests regarding these variables are forwarded to the **`Self Service Backend`**.

umc/self-service/account-verification/email/webserver_address

Defines the `host` part to use in the verification link URL. The default is to use the FQDN of the **`Self Service Backend`** defined via the Univention Configuration Registry Variable *self-service/backend-server* (page 304) since this Univention Configuration Registry Variable is evaluated there.

umc/self-service/account-verification/email/sender_address

Defines the sender address of the verification email. Default is `Account Verification Service <noreply@`*`FQDN`*`>`.

umc/self-service/account-verification/email/server

Server name or IP address of the mail server to use.

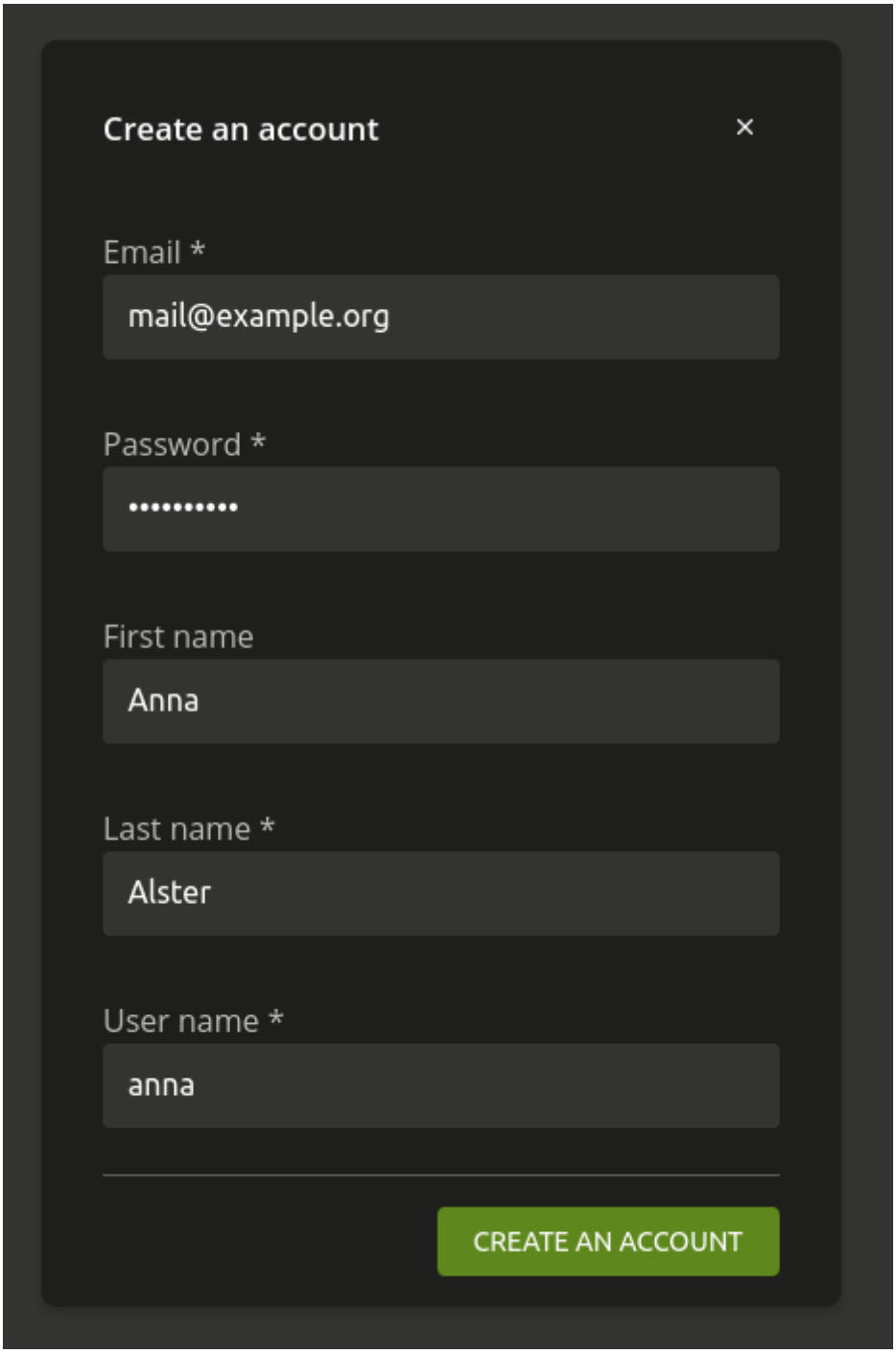


Fig. 6.9: Account registration

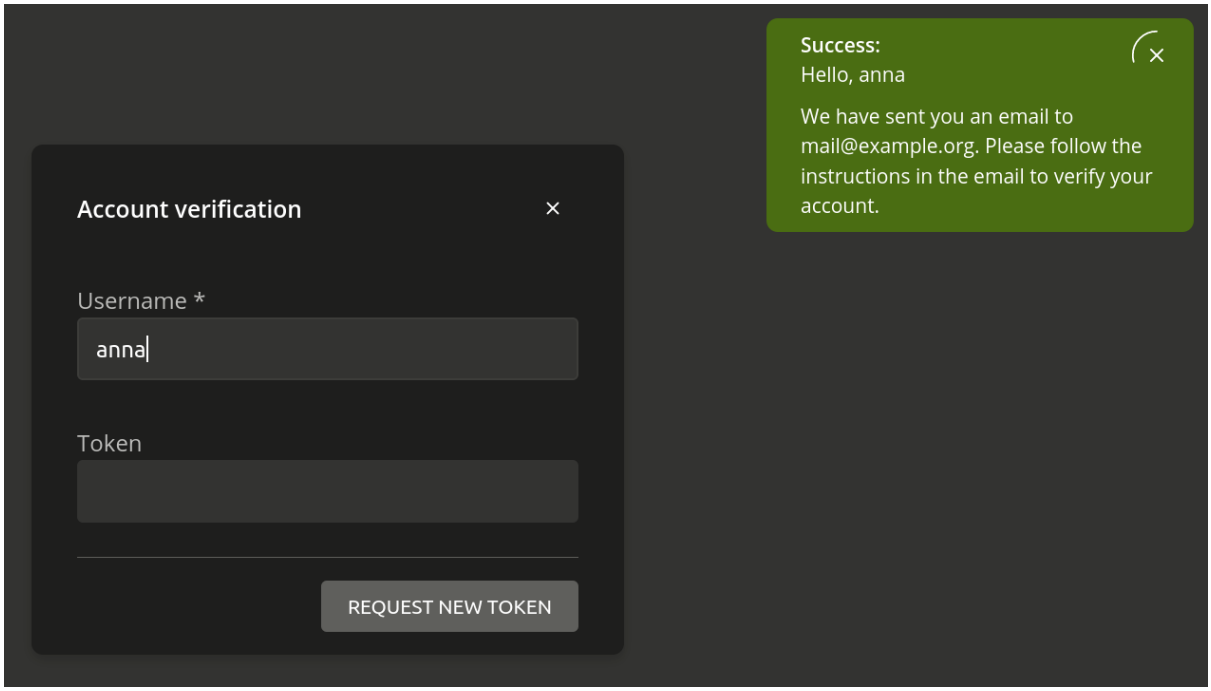


Fig. 6.10: Sending the verification email

umc/self-service/account-verification/email/text_file

A path to a text file whose content will be used for the body of the verification email. The text can contain the following strings which will be substituted accordingly: `{link}`, `{token}`, `{tokenlink}` and `{username}`. As default the file `/usr/share/univention-self-service/email_bodies/verification_email_body.txt` is used.

umc/self-service/account-verification/email/token_length

Defines the number of characters that is used for the verification token. Defaults to `64`.

Account verification

Following the verification link from the email, the user will land on the *Account verification* page of the **Self Service**.

The account verification and request of new verification tokens can be disabled/enabled with the Univention Configuration Registry Variable *umc/self-service/account-verification/backend/enabled* (page 306). This Univention Configuration Registry Variable has to be set on the systems that is defined as **Self Service Backend** via the Univention Configuration Registry Variable *self-service/backend-server* (page 304).

The SSO login can be configured to deny login from unverified, self registered accounts. This is configured through the Univention Configuration Registry Variable *saml/idp/selfservice/check_email_verification* (page 304). This needs to be set on the Primary Directory Node and all Backup Directory Nodes. The setting has no effect on accounts created by an administrator.

The message on the SSO login page for unverified, self registered accounts, can be set with the Univention Configuration Registry Variables *saml/idp/selfservice/account-verification/error-title* (page 304) and *saml/idp/selfservice/account-verification/error-descr* (page 304). A localized message can be configured by adding a *locale* like `en` to the variable, for example `saml/idp/selfservice/account-verification/error-title/`*en*.

If the **Keycloak** app is used as identity provider see Settings[27] in the *Univention Keycloak app documentation* [4] for the corresponding settings.

[27] https://docs.software-univention.de/keycloak-app/latest/configuration.html#app-settings

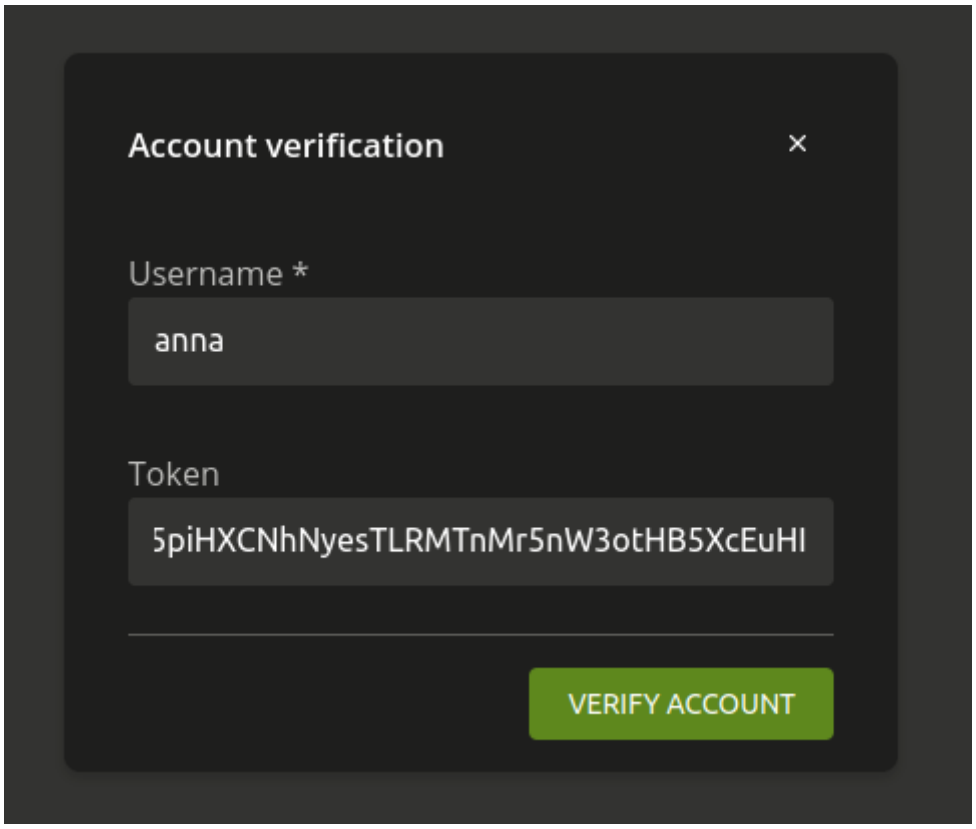


Fig. 6.11: Account verification

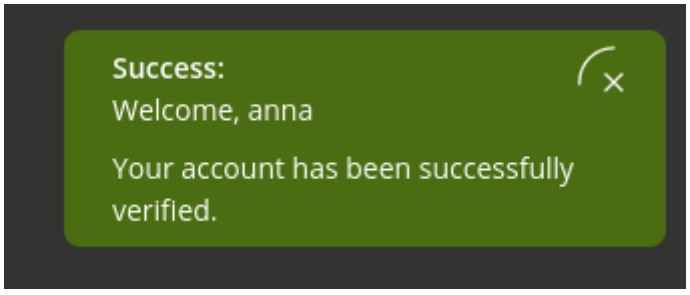


Fig. 6.12: Account verification message

6.5.5 Self deregistration

The **Self Service** allows for users to request the deletion of their own account. This feature can be activated with the Univention Configuration Registry Variable *umc/self-service/account-deregistration/enabled* (page 306), which will show a *Delete my account* Button on the *Your profile* page of the Self Service (*User templates* (page 129)).

If a user has requested to delete their account, it will not be deleted directly but deactivated. In addition the DeregisteredThroughSelfService attribute of the user will be set to TRUE and the DeregistrationTimestamp attribute of the user will be set to the current time in the GeneralizedTime LDAP syntax[28]. If the user has a PasswordRecoveryEmail set they will receive a notification email which can be configured with the following Univention Configuration Registry Variables.

umc/self-service/account-deregistration/email/sender_address

Defines the sender address of the email. Default is Password Reset Service <noreply@*FQDN*>.

umc/self-service/account-deregistration/email/server

Server name or IP address of the mail server to use.

umc/self-service/account-deregistration/email/text_file

A path to a text file whose content will be used for the body of the email. The text can contain the following strings which will be substituted accordingly: {username}. As default the file /usr/share/univention-self-service/email_bodies/deregistration_notification_email_body.txt is used.

The Self Service provides a script under /usr/share/univention-self-service/delete_deregistered_accounts.py that can be used to delete all users/user objects which have DeregisteredThroughSelfService set to TRUE and whose DeregistrationTimestamp is older than a specified time.

The following command would delete users whose DeregistrationTimestamp is older than 5 days and 2 hours:

```
$ /usr/share/univention-self-service/delete_deregistered_accounts.py \
  --timedelta-days 5 \
  --timedelta-hours 2
```

For all possible arguments to the script see:

```
$ /usr/share/univention-self-service/delete_deregistered_accounts.py --help
```

The script can be run regularly by creating a cron job via Univention Configuration Registry.

```
$ ucr set cron/delete_deregistered_accounts/command=\
/usr/share/univention-self-service/delete_deregistered_accounts.py\
' --timedelta-days 30'\
  cron/delete_deregistered_accounts/time='00 06 * * *'  # daily at 06:00
```

More information on how to set cron jobs via Univention Configuration Registry can be found in *Defining cron jobs in Univention Configuration Registry* (page 167).

[28] https://ldapwiki.com/wiki/Wiki.jsp?page=GeneralizedTime

6.6 Automatic lockout of users after failed login attempts

By default, a user can enter their password incorrectly any number of times. To hinder brute force attacks on passwords, an automatic lockout for user accounts can be activated after a configured number of failed login attempts.

UCS unifies various methods for user authentication and authorization. Depending on the installed software components, there may be different mechanisms for configuring and counting failed login attempts.

The three different methods are described in the next sections.

6.6.1 Samba Active Directory Service

In Samba Active Directory environments, various services are provided by Samba, such as Kerberos. To lockout users after too many failed login attempts, the tool **samba-tool** can be used.

- To show the currently configured values:

```
$ samba-tool domain passwordsettings show
```

- To specify how often a user can attempt to sign in with an incorrect password before the account is locked:

```
$ samba-tool domain passwordsettings set --account-lockout-threshold=5
```

- To specify the number of minutes an account will be locked after too many incorrect passwords have been entered:

```
$ samba-tool domain passwordsettings set --account-lockout-duration=3
```

- To define the number of minutes after which the counter is reset:

```
$ samba-tool domain passwordsettings set --reset-account-lockout-after=5
```

  If an account gets automatically unlocked after the lockout duration, the counter is not reset immediately, to keep the account under strict monitoring for some time. During the time window between the end of the lockout duration and the point when the counter gets reset, a single attempt to sign in with an incorrect password will lock the account immediately again.

The manual unlocking of a user is done in the user administration on the tab *Account* by activating the checkbox *Unlock account*.

6.6.2 PAM-Stack

The automatic locking of users after failed logins in the PAM stack can be enabled by setting the Univention Configuration Registry Variable *auth/faillog* (page 291) to `yes`. The upper limit of failed login attempts at which an account lockout is configured in the Univention Configuration Registry Variable *auth/faillog/limit* (page 291). The counter is reset each time the password is entered correctly.

The lockout is activated locally per system by default. In other words, if a user enters their password incorrectly too many times on one system, they can still login on another system. Setting the Univention Configuration Registry Variable *auth/faillog/lock_global* (page 291) will make the lock effective globally and register it in the LDAP directory. The global lock can only be set on Primary Directory Node/Backup systems as other system roles do not have the necessary permissions in the LDAP directory. On all systems with any of these system roles, the lockout gets automatically activated locally or deactivated again via the listener module, depending on the current lock state in the LDAP directory.

As standard, the lockout is not subject to time limitations and must be reset by the administrator. However, it can also be reset automatically after a certain time interval has elapsed. This is done by specifying a time period in seconds in the Univention Configuration Registry Variable *auth/faillog/unlock_time* (page 291). If the value is set to `0`, the lock is reset immediately.

By default, the `root` user is excluded from the password lock, but can also be subjected to it by setting the Univention Configuration Registry Variable *auth/faillog/root* (page 291) to `yes`.

If accounts are only locked locally, the administrator can unlock a user account by entering the command:

```
$ faillog -r -u USERNAME
```

If the lock occurs globally in the LDAP directory, the user can be reset in the Univention Management Console module *Users* on the tab *Account* via the checkbox *Unlock account*.

6.6.3 OpenLDAP

On UCS Directory Nodes, automatic account locking can be enabled for too many failed LDAP server login attempts. The MDB LDAP backend must be used. This is the default backend since UCS 4, previous systems must be migrated to the MDB LDAP backend, see *UCS performance guide* [5].

Automatic account locking must be enabled for each UCS Directory Node. To do this, the Univention Configuration Registry Variables *ldap/ppolicy* (page 296) and *ldap/ppolicy/enabled* (page 297) must be set to `yes` and the OpenLDAP server must be restarted:

```
$ ucr set ldap/ppolicy=yes ldap/ppolicy/enabled=yes
$ systemctl restart slapd
```

The default policy is designed so that five repeated failed LDAP server login attempts within five minutes cause the lockout. A locked account can only be unlocked by a domain administrator through the UMC module *Users* via the checkbox *Unlock account* on the *Account* tab.

The number of repeated failed LDAP server login attempts can be adjusted in the configuration object with the *objectClass* `pwdPolicy`:

```
$ univention-ldapsearch objectclass=pwdPolicy
```

pwdMaxFailure
: attribute determines the number of LDAP authentication errors before locking.

pwdMaxFailureCountInterval
: attribute determines the time interval in seconds that is considered. Failed login attempts outside this interval are ignored in the count.

The following command can be used to block the account after 10 attempts:

```
$ LB="$(ucr get ldap/base)"
$ ldapmodify -x -D "cn=admin,$LB" -y /etc/ldap.secret <<__EOT__
dn: cn=default,cn=ppolicy,cn=univention,$LB
changetype: modify
replace: pwdMaxFailure
pwdMaxFailure: 10
__EOT__
```

The manual unlocking of a user is done in the user administration on the tab *Account* by activating the checkbox *Unlock account*.

6.7 User templates

A user template can be used to preset settings when creating a user. If at least one user template is defined, it can be selected when creating a user.

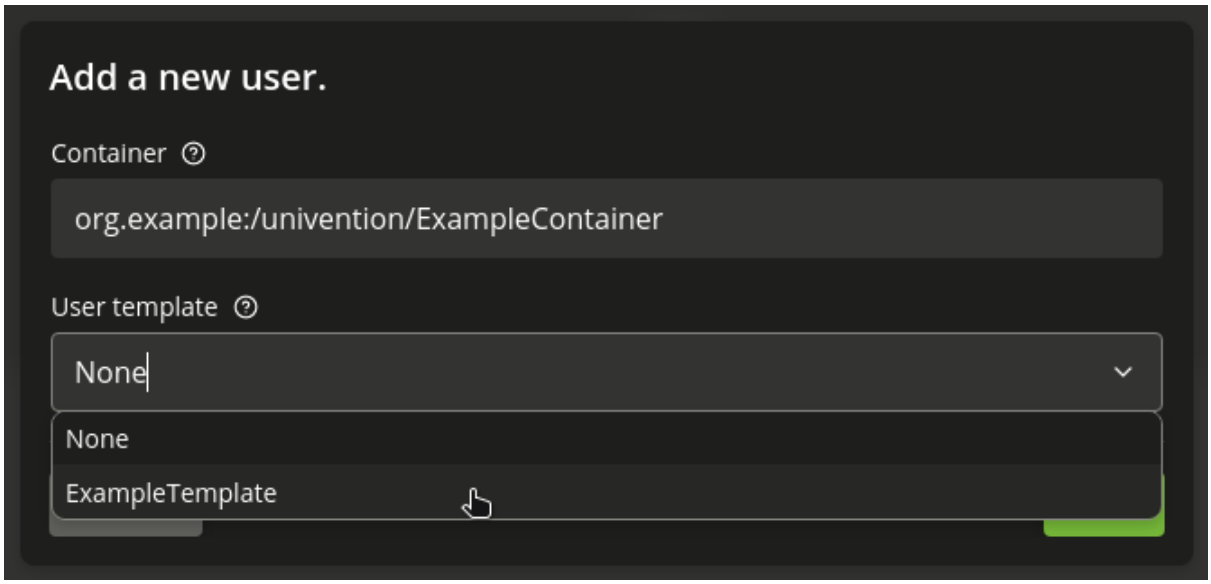


Fig. 6.13: Selecting a user template

User templates are administrated in the UMC module *LDAP directory*. There one needs to switch to the `univention` container and then to the `templates` subcontainer. A new user template can be created here via the *Add* with the object type `Settings: User template`.

In a user template, either a fixed value can be specified (e.g., for the address) or an attribute of the user management referenced. Attributes are then referenced in chevrons.

A list of possible attributes can be displayed with the following command in the section *users/user variables* of the output:

```
$ univention-director-manager users/user
```

If a user template is used for adding a user, this template will overwrite all the fields with the preset values of the template. In doing so, an empty field is set to `""`.

It is also possible to only use partial values of attributes or convert values in uppercase/lowercase.

For example, the UNIX home directory can be stored under `/home/<title>.<lastname>` or the primary email address can be predefined with `<firstname>.<lastname>@company.com`. Substitutions are generally possibly for any value, but there is no syntax or semantics check. So, if no first name is specified when creating a user, the above e-mail address would begin with a dot and would thus be invalid according to the e-mail standard. Similar sources of error can also occur when handling file paths etc. Non-resolvable attributes (for instance due to typing errors in the template) are deleted.

If only a single character of an attribute is required instead of the complete attribute value, the index of the required character can be entered in the user template in square parentheses after the name of the attribute. The count of characters of the attribute begins with `0`, so that index `1` corresponds to the second character of the attribute value. Accordingly, `<firstname>[0].<lastname>@company.com` means an e-mail address will consist of the first letter of the first name plus the last name.

A sub string of the attribute value can be defined by entering a range in square parentheses. In doing so, the index of the first required character and the index of the last required character plus one are to be entered. For example, the input `<firstname>[2:5]` returns the third to fifth character of the first name.

Adding `:lower` or `:upper` to the attribute name converts the attribute value to lowercase or uppercase, e.g.,

<firstname:lower>. If a modifier like :lower is appended to the entire field, the complete value is transformed, e.g. <lastname>@company.com<:lower>.

The option :umlauts can be used to convert special characters such as *è*, *ä* or *ß* into the corresponding ASCII characters.

The option :alphanum can be used to remove all non alphanumeric characters such as ` (backtick) or # (hash). A whitelist of characters that are ignored by this option can be defined in the UCR variable *directory/manager/templates/alphanum/whitelist* (page 292). If this option is applied to an entire field, even manually placed symbols like the @ in an email address are removed. To avoid that, this option should be applied to specific attributes only or needed symbols should be entered into the whitelist.

The options :strip or :trim remove all white space characters from the start and end of the string.

It is also possible to combine options, e.g: :umlauts,upper.

6.8 Overlay module for recording an account's last successful LDAP bind

Caution: Before using this feature please read KB 14404 - support article about activating the OpenLDAP lastbind overlay module[29].

The optional lastbind overlay module[30] for OpenLDAP allows recording the timestamp of the last successful LDAP bind in the authTimestamp attribute and can for example be used to detect unused accounts.

The lastbind overlay module can be activated by setting the Univention Configuration Registry Variable *ldap/overlay/lastbind* (page 296) to yes and restarting the OpenLDAP server. When the module is activated on an UCS server, a timestamp is written to the account's authTimestamp attribute when that account logs into the LDAP server. The Univention Configuration Registry Variable *ldap/overlay/lastbind/precision* (page 296) can be used to configure the time in seconds that has to pass before the authTimestamp attribute is updated. This prevents a large number of write operations that can impair performance.

The authTimestamp attribute can only be queried on the LDAP server where the lastbind overlay module is activated. It is not replicated to other LDAP servers. For that reason the /usr/share/univention-ldap/univention_lastbind.py script can be executed to collect the youngest authTimestamp value from all reachable LDAP servers in the UCS domain and save it into the lastbind extended UDM attribute of a user. The script can be invoked to update the lastbind extended attribute of one or all users. The lastbind extended attribute maps to the univentionAuthTimestamp LDAP attribute.

One way to keep the lastbind extended attribute up-to-date is by creating a cron job via UCR:

```
$ ucr set cron/update_lastbind_attribute/command='\
/usr/share/univention-ldap/univention_lastbind.py --allusers'\
  cron/update_lastbind_attribute/time='00 06 * * *'  # daily at 06:00 a.m.
```

More information on how to set cron jobs via UCR can be found in *Defining cron jobs in Univention Configuration Registry* (page 167).

[29] https://help.univention.com/t/14404
[30] https://manpages.ubuntu.com/manpages/xenial/man5/slapo-lastbind.5.html

6.9 Prevent reuse of user property values

New in version 5.0-6-erratum-974: Since UCS 5.0 erratum 974[31], UCS supports block lists to prevent the reuse of user or group property values.

Block lists is a module in UDM. It allows to configure block lists for UDM properties. When an administrator or software modifies or removes a UDM property on a UDM object, the block list automatically adds an entry about this property with its value to the block list. The entry in the block list prevents that another UDM object can use the same value of the UDM property. Block lists operate on the UDM level.

For example, you want to prevent that UCS reuses values of the UDM property `mailPrimaryAddress` of the UDM objects *user*. You configure a block list for the UDM property `mailPrimaryAddress`. If you then remove the value `chef@example.com` for the UDM property `mailPrimaryAddress` from a UDM user object, the UDM block list creates an entry for that value. If you change the value from `james@example.com` to `john@example.com` for the UDM property `mailPrimaryAddress`, the UDM block list creates another entries for `james@example.com`.

UDM block lists now prevents reusing the values `chef@example.com` and `james@example.com`. You can't use them on other UDM user objects for the UDM property `mailPrimaryAddress`.

6.9.1 Activate block lists

Before you can activate the block lists, you first need to update the UCS systems, where you manage UDM objects, to at least UCS 5.0 erratum 974[32].

Second, you need to set the Univention Configuration Registry Variable *directory/manager/blocklist/enabled* (page 292) to `true` with *ucr set* (page 158) on all UCS systems, where you manage UDM objects.

6.9.2 Configure block lists

You can create, list, and remove block list entries in the UMC module *Blocklists*, or through the command line tool **udm blocklists/list**.

On every block list you need to define the following properties:

Name
: Provides a human-readable name for the block list for later identification.

Retention time
: Defines the retention time for entries in this block list. The retention time is the time period that needs to expire to automatically remove entries from the block list. For example `1m 20d` which results in one month and twenty days.

Properties to block
: Defines the UDM modules and their properties that the block list prevents from reuse.

The following example for **udm blocklists/list** shows how to create a block list through the command line. The block list prevents the reuse of the UDM property `mailPrimaryAddress` for `users/user` objects and the UDM property `mailAddress` for `groups/group` objects.

```
$ udm blocklists/list create \
  --set name=user-and-group-emails \
  --set retentionTime=40d \
  --append blockingProperties="users/user mailPrimaryAddress" \
  --append blockingProperties="groups/group mailAddress"
```

[31] https://errata.software-univention.de/#/?erratum=5.0x974
[32] https://errata.software-univention.de/#/?erratum=5.0x974

6.9.3 Manage block list entries

You can manage block list entries in the UMC module *Blocklists*, or through the command line tool **udm blocklists/list**.

When you activated block lists, UDM automatically creates entries in the configured block list, when you remove a value from a UDM property of a UDM object. UDM automatically deletes expired entries from the block list.

Every block list entry has the following properties:

Value
: A SHA-256 hash representing the value that the block list is blocking from reuse. The UDM property value is a clear text value. Before UDM creates the block list entry, it converts the value to lowercase text. All uppercase and lowercase variants of the value then match the block list entry when validated by UDM.

Blocked until
: The timestamp when the block list entry expires. It uses the GeneralizedTime-LDAP-Syntax[33] format.

When UDM creates a block list entry, it takes the current date and time, adds the configured retention time of the corresponding block list and writes the result to *Blocked until*.

Changing the retention time of the block list doesn't update the *Blocked until* property of the block list entry.

Origin ID
: The ID of the UDM object that caused the block list entry. You can still use the value of the block list entry on this UDM object.

Important: Listing block list entries gives you only the hashes of the blocked values.

Nevertheless, you can search for the clear text value of a particular entry, for example, in case you want to delete that entry.

```
$ udm blocklists/entry list
DN: cn=sha256:a859cd5964b6ac...,cn=emails,cn=blocklists
DN: cn=sha256:b859cd5964b6ac...,cn=emails,cn=blocklists
DN: cn=sha256:c859cd5964b6ac...,cn=emails,cn=blocklists

$ udm blocklists/entry list --filter value=blocked_email@example.com
DN: cn=sha256:c859cd5964b6ac...,cn=emails,cn=blocklists
```

6.9.4 Expired block list entries

Every entry in a block list has a *Blocked until* property. Block list entries expire after the timestamp passed. A cron job on the Primary Directory Node deletes expired block list entries.

You can configure how often cron runs the job with the Univention Configuration Registry Variable *directory/manager/blocklist/cleanup/cron* (page 292). The log file `/var/log/univention/blocklist-clean-expired-entries.log` lists the expired entries that UDM deleted.

[33] https://ldapwiki.com/wiki/Wiki.jsp?page=GeneralizedTime

6.9.5 LDAP ACLs for block lists

By default every UCS node in the domain and every member of the `Domain Admins` group can write block list entries. And everybody can read. You can configure the permissions on the Primary Directory Node and the Backup Directory Nodes with the following Univention Configuration Registry Variables:

- *ldap/database/internal/acl/blocklists/groups/read* (page 295)
- *ldap/database/internal/acl/blocklists/groups/write* (page 295)

For example, if you want to give a user the permission to delete block list entries who isn't member of the `Domain Admins` group, you need to create a group with that user as member and add the LDAP DN of this group to *ldap/database/internal/acl/blocklists/groups/write* (page 295).

CHAPTER

SEVEN

GROUP MANAGEMENT

Permissions in UCS are predominantly differentiated between on the basis of *groups*. Groups are stored in the LDAP and are thus identical on all systems. Groups can contain not only user accounts, but can also optionally accept computer accounts.

In addition, there are also local user groups on each system, which are predominantly used for hardware access. These are not managed through the UCS Management System, but saved in the `/etc/group` file.

Note: The group management is part of Univention Nubus in the *Directory Manager* component. For more information about Nubus, refer to *What is Univention Nubus?* (page 2)

7.1 User group assignments

The assignment of users to groups is performed in two ways:

- A selection of groups can be assigned to a user in the user management, see *User management through Univention Management Console module* (page 106).
- A selection of users can be assigned to a group in the group management, see *Managing groups via Univention Management Console module* (page 136).

7.2 Recommendation for group name definition

One very important and required attribute for groups is the group name. To avoid conflicts with the different tools handling groups in UCS, adhere to the following recommendations for the definition of group names:

- Only use upper and lower case letters (`A-Za-z`), digits (`0-9`) the hyphen (`-`) and space from the ASCII character set for group names.
- The group name starts with a letter from the ASCII character set. The space is not allowed as first or last character. The hyphen is not allowed as last character.
- In UCS the group name has at least a length of 4 characters and at most 20 characters.

The recommendation results in the following regular expression:

```
^[A-Za-z][A-Za-z0-9 -]{2,18}[A-Za-z0-9]$
```

Consider the recommendation as a guideline and not a rule and keep potential side-effects in mind when defining group names outside the recommendation.

7.3 Managing groups via Univention Management Console module

Groups are managed in the UMC module *Groups* (see *Univention Management Console modules* (page 69)).

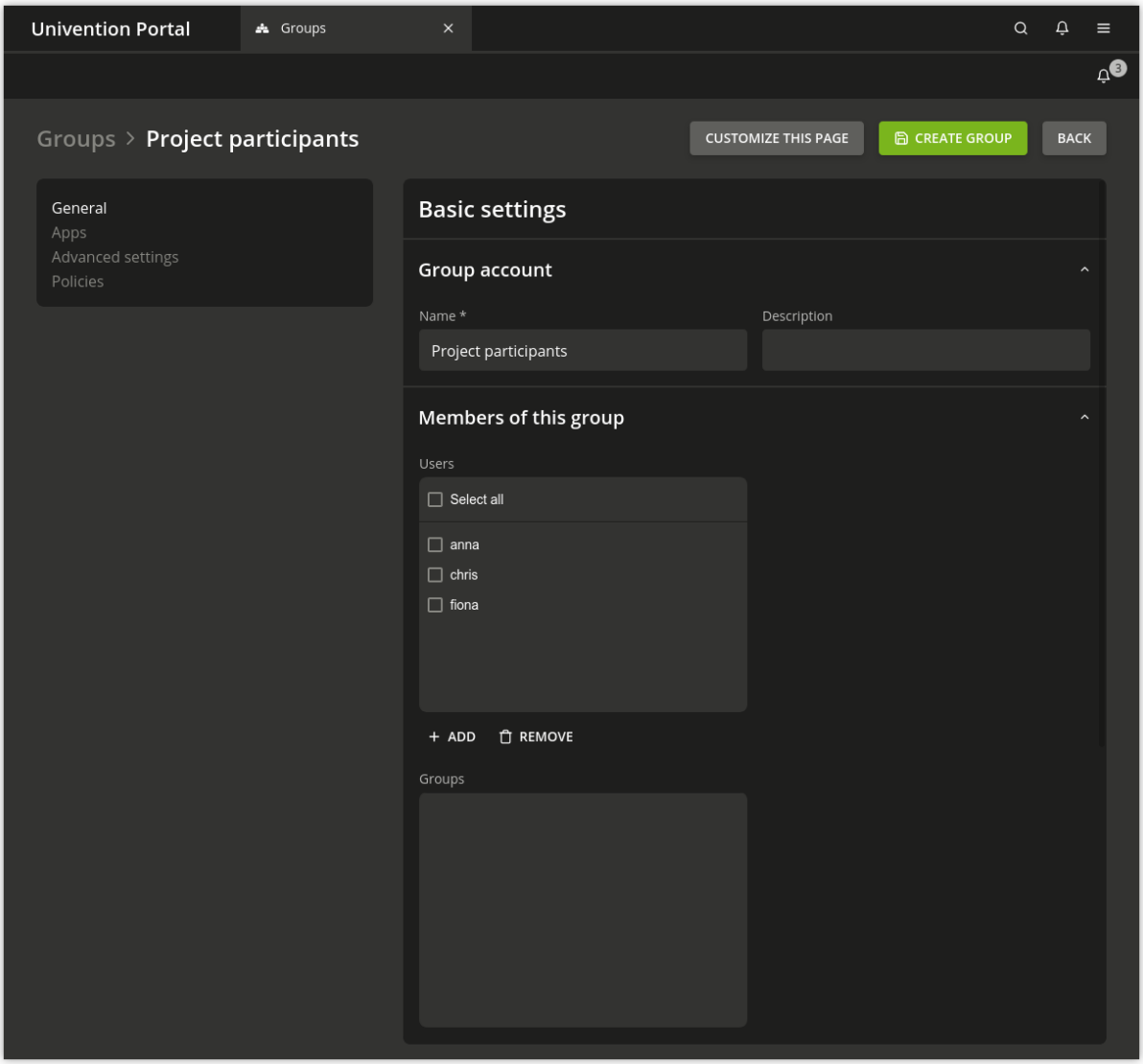


Fig. 7.1: Creating a group via UMC module

7.3.1 Group management module - General tab

Table 7.1: *General* tab

Attribute	Description
Name (*)	Defines the name of the group. For recommended characters for the group name, see *Recommendation for group name definition* (page 135). By default it is not possible to create a group with the same name as an existing user. If the Univention Configuration Registry Variable `directory/manager/user_group/uniqueness` (page 292) is set to `false`, this check is removed.
Description	A description of the group can be entered here.
Users	This input field can be used for adding users as members to the group.
Groups	On this input field, other groups can be added as members of the current group (groups in groups).

7.3.2 Group management module - Advanced settings tab

Table 7.2: *Advanced settings* tab

Attribute	Description
Mail	These options define a mail group and are documented in the *Management of mail groups* (page 264).
Host members	This field can be used for accepting computers as members of the group.
Nested groups	The current group can be added as a member to other groups here (groups in groups).
Group ID	If a group is to be assigned a certain group ID, the ID in question can be entered in this field. Otherwise the next available group ID will be automatically assigned when adding the group. The group ID cannot be subsequently changed. When editing the group, the group ID will be represented in gray. The group ID may consist of integers between 1000 and 59999 and between 65536 and 100000.
Windows ▸ Relative ID	The relative ID (RID) is the local part of the Security ID (SID) and is used in Windows and Samba domains. If a group is to be assigned a certain RID, the ID in question can be entered in this field. Otherwise a RID will be automatically assigned. The RID cannot be subsequently changed. When editing the group, the group ID will be represented in gray. The RIDs below 1000 are reserved for standard groups and other special objects. When Samba/AD is used, the RID is generated by Samba and cannot be specified.
Windows ▸ group type	This group type is evaluated when the user logs in to a Samba/AD-based domain. Three types of Windows groups can be distinguished: **Domain Groups** are known across the domain. This is the default group type. **Local groups** are only relevant on Windows servers. If a local group is created on a Windows server, this group is known solely to the server; it is not available across the domain. UCS, in contrast, does not differentiate between local and global groups. After taking over an AD domain, local groups in UCS can be handled in the same way as global groups. **Well-known group** This group type covers groups preconfigured by Samba/Windows servers which generally have special privileges, e.g., `Power Users`.
Windows ▸ AD group type	This group type is only evaluated when the user logs in to a Samba/AD-based domain (which offers Active Directory domain services). These groups are described in *Synchronization of Active Directory groups when using Samba/AD* (page 140).
Windows ▸ Samba privileges	This input mask can be used to assign Windows system rights to a group, e.g., the right to join a Windows client in the domain. This function is documented in *User management through Univention Management Console module* (page 106).

7.3.3 Group management module - Options settings tab

This tab is only available when adding groups, not when editing groups. Certain LDAP object classes for the group can be de-selected here. The entry fields for the attributes of these classes can then no longer be filled in.

Table 7.3: *Options* tab

Attribute	Description
Samba group	This checkbox indicates whether the group contains the object class `sambaGroupMapping`.
POSIX group	This checkbox indicates whether the group contains the object class `posixGroup`.

7.4 Group nesting with groups in groups

UCS supports group nesting (also known as "groups in groups"). This simplifies the management of the groups. For example, if two locations are managed in one domain, two groups can be formed `IT staff location A` and `IT staff location B`, to which the user accounts of the location's IT staff can be assigned respectively.

To create a cross-location group, it is then sufficient to define the groups `IT staff location A` and `IT staff location B` as members.

Cyclic dependencies of nested groups are automatically detected and refused. This check can be disabled with the Univention Configuration Registry Variable *directory/manager/web/modules/groups/group/checks/circular_dependency* (page 293). Cyclic memberships must also be avoided in direct group changes without the UCS Management System.

The resolution of nested group memberships is performed during the generation of the group cache (see *Local group cache* (page 139)) and is thus transparent for applications.

7.5 Local group cache

The user and computer information retrieved from the LDAP is cached by the Name Server Cache Daemon (NSCD), see *Name service cache daemon* (page 168).

Since UCS 3.1, the groups are no longer cached via the NSCD for performance and stability reasons; instead they are now cached by the NSS module **`libnss-extrausers`**. The group information is automatically exported to the `/var/lib/extrausers/group` file by the `/usr/lib/univention-pam/ldap-group-to-file.py` script and read from there by the NSS module.

In the basic setting, the export is performed once a day by a cron job and is additionally started if the Univention Directory Listener has been inactive for 15 seconds. The interval for the cron update is configured in Cron syntax (see *Defining local cron jobs in /etc/cron.d/* (page 167)) by the Univention Configuration Registry Variable *nss/group/cachefile/invalidate_interval* (page 301). This listener module can be activated/deactivated via the Univention Configuration Registry Variable *nss/group/cachefile/invalidate_on_changes* (page 301) (`true`/`false`).

When the group cache file is being generated, the script can verify whether the group members are still present in the LDAP directory. If not only UMC modules are used for user management, this additional check can be can be enabled by setting the Univention Configuration Registry Variable *nss/group/cachefile/check_member* (page 300) to `true`.

7.6 Synchronization of Active Directory groups when using Samba/AD

If Samba/AD is used, the group memberships are synchronized between the Samba/AD directory service and the OpenLDAP directory service by the Univention S4 connector, i.e., each group on the UCS side is associated with a group in Active Directory. General information on the Univention S4 connector can be found in *Univention S4 connector* (page 173).

Some exceptions are formed by the *pseudo groups*, sometimes also called system groups. These are only managed internally by Active Directory/Samba, e.g., the `Authenticated Users` group includes a list of all the users currently logged on to the system. Pseudo groups are stored in the UCS directory service, but they are not synchronized by the Univention S4 connector and should usually not be edited. This applies to the following groups:

- `Anonymous Logon`
- `Authenticated Users`
- `Batch`
- `Creator Group`
- `Creator Owner`
- `Dialup`
- `Digest Authentication`
- `Enterprise Domain Controllers`
- `Everyone`
- `IUSR`
- `Interactive`
- `Local Service`
- `NTLM Authentication`
- `Network Service`
- `Network`
- `Nobody`
- `Null Authority`
- `Other Organization`
- `Owner Rights`
- `Proxy`
- `Remote Interactive Logon`
- `Restricted`
- `SChannel Authentication`
- `Self`
- `Service`
- `System`
- `Terminal Server User`
- `This Organization`
- `World Authority`

In Active Directory/Samba, a distinction is made between the following four AD group types. These group types can be applied to two types of groups; *security groups* configure permissions (corresponding to the UCS groups), whilst *distribution groups* are used for mailing lists:

Local
: *Local* groups only exist locally on a host. A local group created in Samba/AD is synchronized by the Univention S4 Connector and thus also appears in the UMC module *Groups*. There is no need to create local groups in the UMC module.

Global
: *Global* groups are the standard type for newly created groups in the UMC module *Groups*. A global group applies for one domain, but it can also accept members from other domains. If there is a trust relationship with a domain, the groups there are displayed and permissions can be assigned. However, the current version of UCS does not support multiple domains/forests or outgoing trust relationships.

Domain local
: *Domain local* groups can also adopt members of other domains (insofar as there is a trust relationship in place or they form part of a forest). Local domain groups are only shown in their own domain though. However, the current version of UCS does not support multiple domains/forests or outgoing trust relationships.

Universal
: *Universal* groups can adopt members from all domains and these members are also shown in all the domains of a forest. These groups are stored in a separate segment of the directory service, the so-called *global catalog*. Domain forests are currently not supported by Samba/AD.

7.7 Overlay module for displaying the group information on user objects

In the UCS directory service, group membership properties are only saved in the group objects and not in the respective user objects. However, some applications expect group membership properties at the user objects (e.g., in the attribute `memberOf`). An optional overlay module in the LDAP server makes it possible to present these attributes automatically based on the group information. The additional attributes are not written to the LDAP, but displayed on the fly by the overlay module if a user object is queried.

Caution: Before using this feature, please read KB 6439 - memberOf attribute: Group memberships of user and computer objects[34] about activating the OpenLDAP `memberOf` overlay module.

To this end, the **univention-ldap-overlay-memberof** package must be installed on all LDAP servers. Afterwards **/usr/share/univention-ldap-overlay-memberof/univention-update-memberof** must be invoked on all servers.

By default the user attribute `memberOf` is shown. The Univention Configuration Registry Variable *ldap/overlay/memberof/memberof* (page 296) can be used to configure a different attribute.

[34] https://help.univention.com/t/6439

CHAPTER

EIGHT

COMPUTER MANAGEMENT

8.1 Management of computer accounts via Univention Management Console module

All UCS, Linux and Windows systems within a UCS domain each have a computer domain account (also referred to as the host account) with which the systems can authenticate themselves among each other and with which they can access the LDAP directory.

The computer account is generally created automatically when the system joins the UCS domain (see *Joining domains* (page 29)); however, the computer account can also be added prior to the domain join.

The password for the computer account is generated automatically during the domain join and saved in the `/etc/machine.secret` file. By default the password consists of 20 characters (can be configured via the Univention Configuration Registry Variable *machine/password/length* (page 297)). The password is regenerated automatically at fixed intervals (default setting: 21 days; can be configured using the Univention Configuration Registry Variable *server/password/interval* (page 304)). Password rotation can also be disabled using the variable *server/password/change* (page 304).

There is an different computer object type for every system role. Further information on the individual system roles can be found in *UCS system roles* (page 33).

Computer accounts are managed in the UMC module *Computers*.

By default a simplified wizard for creating a computer is shown, which only requests the most important settings. All attributes can be shown by clicking on *Advanced*. If there is a DNS forward zone and/or a DNS reverse zone (see *Administration of DNS data with BIND* (page 212)) assigned to the selected network object (see *Network objects* (page 211)), a host record and/or pointer record is automatically created for the host. If there is a DHCP service configured for the network object and a MAC address is configured, a DHCP host entry is created (see *IP assignment via DHCP* (page 220)).

The simplified wizard can be disabled for all system roles by setting the Univention Configuration Registry Variable *directory/manager/web/modules/computers/computer/wizard/disabled* (page 292) to `true`.

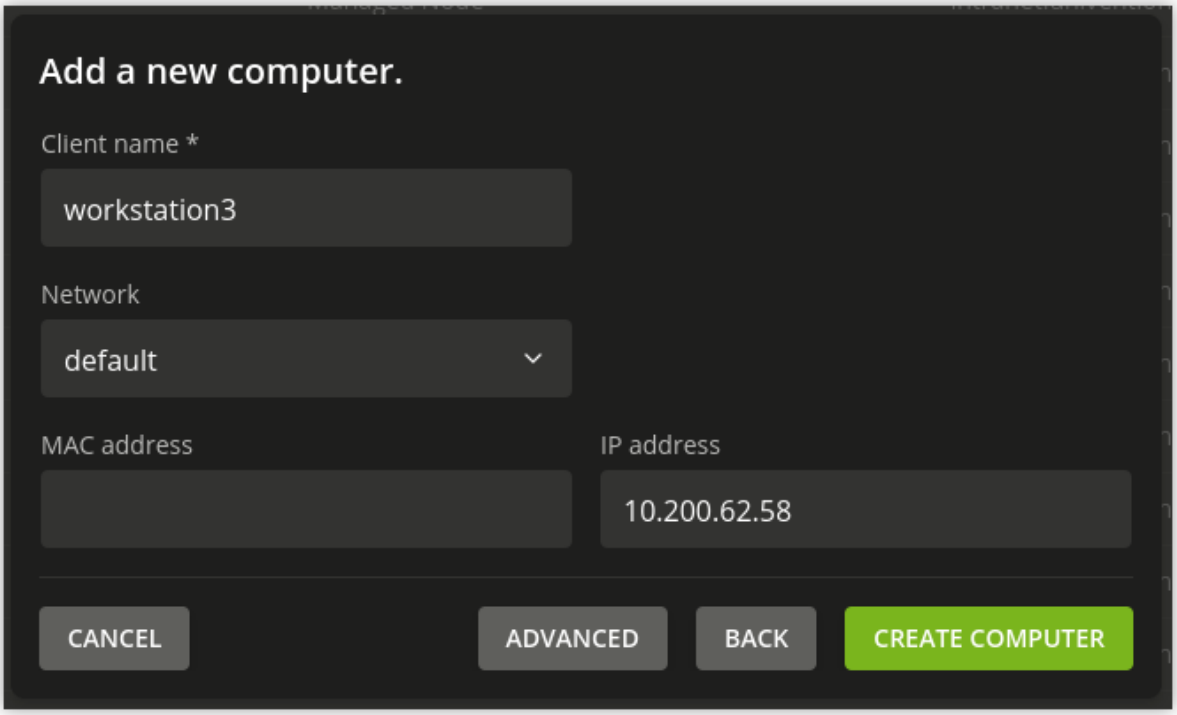


Fig. 8.1: Creating a computer in the UMC module

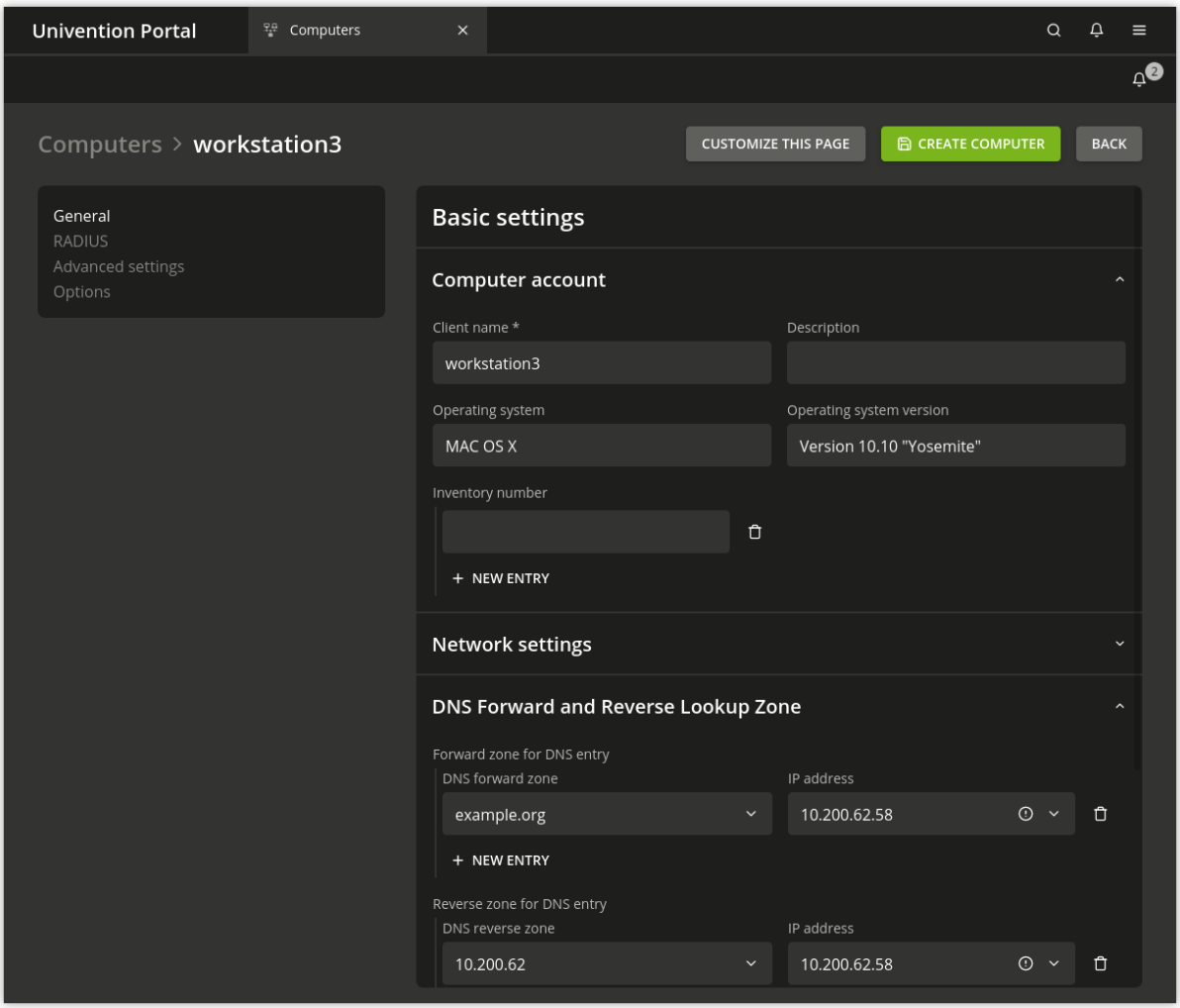


Fig. 8.2: Advanced computer settings

8.1.1 Computer management module - General tab

Table 8.1: *General* tab

Attribute	Description
Name	The name for the host should be entered in this input field. To guarantee compatibility with different operating systems and services, computer names should only contain the lowercase letters *a* to *z*, numbers, hyphens and underscores. Umlauts and special characters are not permitted. The full stop is used as a separating mark between the individual components of a fully qualified domain name and must therefore not appear as part of the computer name. Computer names must begin with a letter. Microsoft Windows accepts computer names with a maximum of 13 characters, so as a rule computer names should be limited to 13 characters if there is any chance that Microsoft Windows will be used. After creation, the computer name can only be changed for the system roles *Windows Workstation/Server*, *macOS Client* and *IP client*.
Description	Any description can be entered for the host in this input field.
Inventory number	Inventory numbers for hosts can be stored here.
Network	The host can be assigned to an existing network object. Information on the IP configuration can be found in *Network objects* (page 211).
MAC address	The MAC address of the computer can be entered here, for example `2e:44:56:3f:12:32`. If the computer is to receive a DHCP entry, the entry of the MAC address is essential.
IP address	Fixed IP addresses for the host can be given here. Further information on the IP configuration can be found in *Network objects* (page 211). If a network was selected on the *General* tab, the IP address assigned to the host from the network will be shown here automatically. An IP address entered here (i.e. in the LDAP directory) can only be transferred to the host via DHCP. If no DHCP is being used, the IP address must be configured locally, see *Network configuration* (page 151). If the IP addresses entered for a host are changed without the DNS zones being changed, they are automatically changed in the computer object and - where they exist - in the DNS entries of the forward and reverse lookup zones. If the IP address of the host was entered at other places as well, these entries must be changed manually! For example, if the IP address was given in a DHCP boot policy instead of the name of the boot server, this IP address will need to be changed manually by editing the policy.
Forward zone for DNS entry	The DNS forward zone in which the computer is entered. The zone is used for the resolution of the computer name in the assigned IP address. Further information on the IP configuration can be found in *Network objects* (page 211).
Reverse zone for DNS entry	The DNS reverse zone in which the computer is entered. The zone is used to resolve the computer's IP address in a computer name. Further information on the IP configuration can be found in *Network objects* (page 211).
DHCP service	If a computer is supposed to procure its IP address via DHCP, a DHCP service must be assigned here. Information on the IP configuration can be found in *Network objects* (page 211). During assignment, it must be ensured that the DHCP servers of the DHCP service object are responsible for the physical network. If a network is selected on the *General* tab an appropriate entry for the network will be added automatically. It can be adapted subsequently.

8.1.2 Computer management module - Account tab

Table 8.2: *Account* tab (advanced settings)

Attribute	Description
Password	The password for the computer account is usually automatically created and rotated. For special cases such as the integration of external systems it can also be explicitly configured in this field. The same password must then also be entered locally on the computer in the `/etc/machine.secret` file.
Primary group	The primary group of the host can be selected in this selection field. This is only necessary when they deviate from the automatically created default values. The default value for a Primary Directory Node or Backup Directory Node is `DC Backup Hosts`, for a Replica Directory Node `DC Slave Hosts` and for Managed Nodes `Computers`.

8.1.3 Computer management module - Unix account tab

Table 8.3: *Unix account* tab (advanced settings)

Attribute	Description
Unix home directory (*)	A different input field for the host account can be entered here. The automatically created default value for the home directory is `/dev/null`.
Login shell	If a different login shell from the default value is to be used for the computer account, the login shell can be adapted manually in this input field. The automatically set default value assumes a login shell of `/bin/sh`.

8.1.4 Computer management module - Services tab

Table 8.4: *Services* tab (advanced settings)

Attribute	Description
Service	By means of a service object, applications or services can determine whether a service is available on a computer or generally in the domain.

Note: The tab *Services* is only displayed on UCS server system roles.

8.1.5 Computer management module - Deployment tab

This *Deployment* tab is used for the Univention Net Installer, see *Extended installation documentation* [6].

8.1.6 Computer management module - DNS alias tab

Table 8.5: *DNS alias* tab (advanced settings)

Attribute	Description
Zone for DNS Alias	If a zone entry for forward mapping has been set up for the host in the *Forward zone for DNS entry* field, the additional alias entries via which the host can be reached can be configured here.

8.1.7 Computer management module - Alerts tab

Table 8.6: *Alerts* tab (advanced settings)

Attribute	Description
Assigned monitoring alerts	Specifies which Monitoring alert checks should be performed for this computer, see *Configure monitoring alerts* (page 285).

8.1.8 Computer management module - Groups tab

The computer can be added into different groups in *Groups* tab.

8.1.9 Computer management module - Options alias tab

The *Options* tab allows to disable LDAP object classes for host objects. The entry fields for attributes of disabled object classes are no longer shown. Not all object classes can be modified subsequently.

Table 8.7: *(Options)* tab

Attribute	Description
Kerberos principal	If this checkbox is not selected the host does not receive the `krb5Principal` and `krb5KDCEntry` object classes.
POSIX account	If this checkbox is not selected the host does not receive the `posixAccount` object class.
Samba account	If this checkbox is not selected the host does not receive the `sambaSamAccount` object class.

8.1.10 Integration of Ubuntu clients

Ubuntu clients can be managed in the UMC module *Computers* with their own system role. The network properties for DNS/DHCP can also be managed there.

The use of policies is not supported.

Some configuration adjustments need to be performed on Ubuntu systems; these are documented in *Extended domain services documentation* [2].

8.2 Configuration of hardware and drivers

8.2.1 Available kernel variants

The standard kernel in UCS 5.0 is based on the Linux kernel 4.19. In principle, there are three different types of kernel packages:

- A *kernel image package* provides an executable kernel which can be installed and started.
- A *kernel source package* provides the source code for a kernel. From this source, a tailor-made kernel can be created, and functions can be activated or deactivated.
- A *kernel header package* provides interface information which is required by external packages if these have to access kernel functions. This information is usually necessary for compiling external kernel drivers.

Normally, the operation of a UCS system only requires the installation of one kernel image package.

Several kernel versions can be installed in parallel. This makes sure that there is always an older version available to which can be reverted in case of an error. So-called meta packages are available which always refer to the kernel version currently recommended for UCS. In case of an update, the new kernel version will be installed, making it possible to keep the system up to date at any time.

8.2.2 Hardware drivers / kernel modules

The boot process occurs in two steps using an initial RAM disk (*initrd* for short). This is composed of an archive with further drivers and programs.

The GRUB boot manager (see *GRUB boot manager* (page 150)) loads the kernel and the *initrd* into the system memory, where the *initrd* archive is extracted and mounted as a temporary root file system. The real root file system is then mounted from this, before the temporary archive is removed and the system start implemented.

The drivers to be used are recognized automatically during system start and loaded via the **udev** device manager. At this point, the necessary device links are also created under `/dev/`. If drivers are not recognized (which can occur if no respective hardware IDs are registered or hardware is employed which cannot be recognized automatically, e.g., ISA boards), kernel modules to be loaded can be added via Univention Configuration Registry Variable *kernel/modules* (page 295). If more than one kernel module is to be loaded, these must be separated by a semicolon. The Univention Configuration Registry Variable *kernel/blacklist* (page 295) can be used to configure a list of one or more kernel modules for which automatic loading should be prevented. Multiple entries must also be separated by a semicolon.

Unlike other operating systems, the Linux kernel (with very few exceptions) provides all drivers for hardware components from one source. For this reason, it is not normally necessary to install drivers from external sources subsequently.

However, if external drivers or kernel modules are required, they can be integrated via the DKMS framework (Dynamic Kernel Module Support). This provides a standardized interface for kernel sources, which are then built automatically for every installed kernel (insofar as the source package is compatible with the respective kernel). For this to happen, the kernel header package **linux-headers-amd64** must be installed in addition to the **dkms** package. Please note that not all the external kernel modules are compatible with all kernels.

8.2.3 GRUB boot manager

In Univention Corporate Server GNU GRUB 2 is used as the boot manager. GRUB provides a menu which allows the selection of a Linux kernel or another operating system to be booted. GRUB can also access file systems directly and can thus, for example, load another kernel in case of an error.

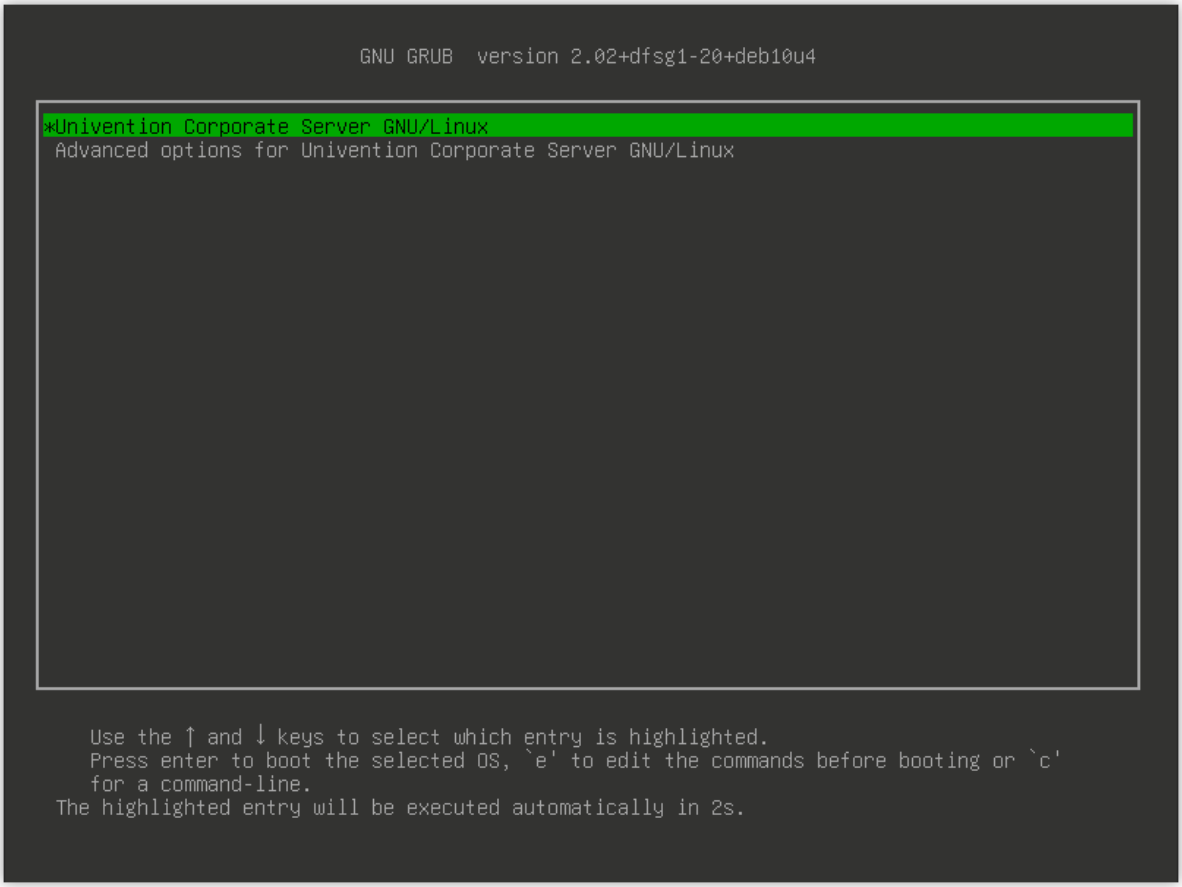


Fig. 8.3: GRUB menu

GRUB gets loaded in a two-step procedure; in the Master Boot Record of the hard drive, the Stage 1 loader is written which refers to the data of Stage 2, which in turn manages the rest of the boot procedure.

The selection of kernels to be started in the boot menu is stored in the file `/boot/grub/grub.cfg`. This file is generated automatically; all installed kernel packages are available for selection. The memory test program **`Memtest86+`** can be started by selecting the option *Memory test* and performs a consistency check for the main memory.

There is a five second waiting period during which the kernel to be booted can be selected. This delay can be changed via the Univention Configuration Registry Variable *`grub/timeout`* (page 294).

By default a screen size of `800x600` pixels and 16 Bit color depth is preset. A different value can be set via the Univention Configuration Registry Variable *`grub/gfxmode`* (page 294). Only resolutions are supported which can be set via VESA BIOS extensions. A list of available modes can be found in VESA BIOS Extensions[35]. The input must be specified in the format *`HORIZONTAL`*`x`*`VERTICAL`*`@`*`COLOURDEPTHBIT`*, so for example `1024x768@16`.

Kernel options for the started Linux kernel can be passed with the Univention Configuration Registry Variable *`grub/append`* (page 294). Univention Configuration Registry Variable *`grub/xenhopt`* (page 294) can be used to pass options to the Xen hypervisor.

The graphic representation of the boot procedure - the so-called splash screen - can be deactivated by setting Univention Configuration Registry Variable *`grub/bootsplash`* (page 294) to `nosplash`.

[35] https://en.wikipedia.org/wiki/VESA_BIOS_Extensions

8.2.4 Network configuration

The configuration of network interfaces can be adjusted with the UMC module *Network settings*.

The configuration is saved in Univention Configuration Registry variables, which can also be set directly. These variables are listed in the individual sections.

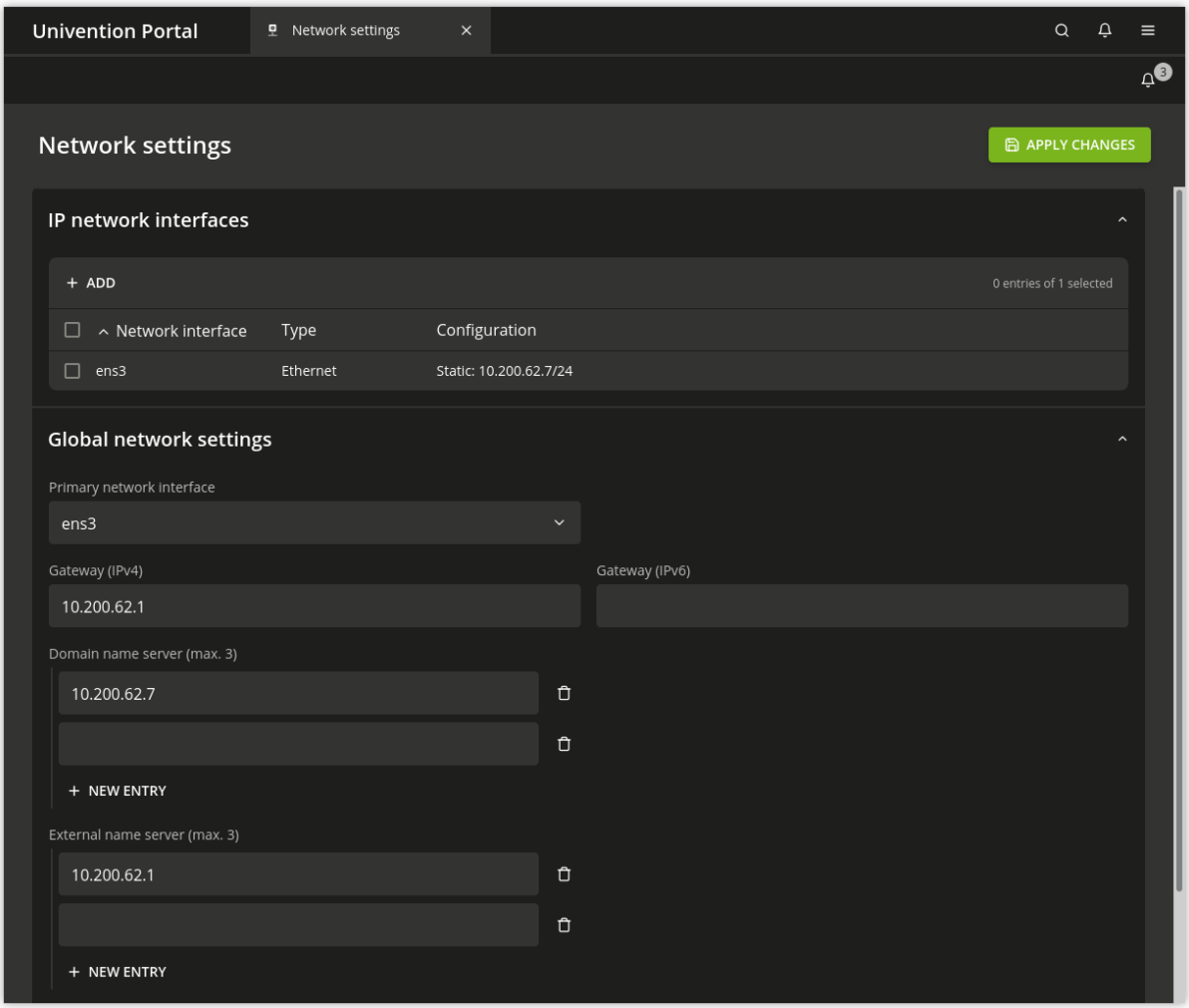


Fig. 8.4: Configuring the network settings

All the network cards available in the system are listed under *IPv4 network devices* and *IPv6 network devices* (only network interfaces in the `ethX` scheme are shown).

Network interfaces can be configured for IPv4 and/or IPv6. IPv4 addresses have a 32-bit length and are generally written in four blocks in decimal form (e.g., `192.0.2.10`), whereas IPv6 addresses are four times as long and typically written in hexadecimal form (e.g., `2001:0DB8:FE29:DE27:0000:0000:0000:0000`).

Configuration of IPv4 addresses

If the *Dynamic (DHCP)* option was not chosen, the IP address to be bound to the network card must be entered. In addition to the *IPv4 address* the *net mask* must also be entered. *DHCP query* is used to request an address from a DHCP server. Unless the *Dynamic (DHCP)* option is activated, the values received from the DHCP request are configured statically.

Server systems can also be configured via DHCP. This is necessary for some cloud providers, for example. If the assignment of an IP address for a server fails, a random link local address (`169.254.x.y`) is configured as a replacement.

For UCS server systems the address received via DHCP is also written to the LDAP directory.

Note: Not all services (e.g., DNS servers) are suitable for use on a DHCP-based server.

UCR variables:

- *interfaces/ethX/address* (page 294)
- *interfaces/ethX/netmask* (page 295)
- *interfaces/ethX/type* (page 295)
- *gateway* (page 294)

Besides the physical interfaces, additional virtual interfaces can also be defined in the form *interfaces/ethX_Y/setting* (page 295).

Configuration of IPv6 addresses

The IPv6 address can be configured in two ways: Stateless address autoconfiguration (SLAAC) is employed in the *Autoconfiguration (SLAAC)* configuration. In this, the IP address is assigned from the routers of the local network segment. Alternatively, the address can also be configured statically by entering the *IPv6 address* and *IPv6 prefix*.

In contrast to DHCP, in SLAAC there is no assignment of additional data such as the DNS server to be used. There is an additional protocol for this (DHCPv6), which, however, is not employed in the dynamic assignment. One network card can be used for different IPv6 addresses. The *Identifier* is a unique name for individual addresses. The main address always uses the identifier `default`; functional identifiers such as `Interface mail server` can be assigned for all other addresses.

UCR variables:

- *interfaces/ethX/ipv6/address* (page 295)
- *interfaces/ethX/ipv6/prefix* (page 295),
- *interfaces/ethX/ipv6/acceptRA* (page 295) activates SLAAC

Further network settings can be performed under *Global network settings*.

The IP addresses for the standard gateways in the subnetwork can be entered under *Gateway (IPv4)* and *Gateway (IPv6)*. It is not obligatory to enter a gateway for IPv6, but recommended. A gateway configured here has preference over router advertisements, which might otherwise be able to change the route.

UCR variables:

- *ipv6/gateway* (page 295)

Configuring the name servers

There are two types of DNS servers:

External DNS Server
An *External DNS Server* is employed for the resolution of host names and addresses outside of the UCS domain, e.g., `univention.de`. This is typically a name server operated by the internet provider.

Domain DNS Server
A *Domain DNS Server* is a local name server in the UCS domain. This name server usually administrates host names and IP addresses belonging to the UCS domain. If an address is not found in the local inventory, an external DNS server is automatically requested. The DNS data are saved in the LDAP directory service, i.e., all domain DNS servers deliver identical data.

A local DNS server is set up on the Primary Directory Node, Backup Directory Node and Replica Directory Node system roles. Here, you can configure which server should be primarily used for the name resolution by entering the *Domain DNS Server*.

UCR variables:

- *nameserver1* (page 300) to *nameserver3* (page 300)
- *dns/forwarder1* (page 293) to *dns/forwarder3* (page 293),

Bridges, bonding, VLANs

UCS supports advanced network configurations using bridging, bonding and virtual networks (VLAN):

- Bridging is often used with virtualization to connect multiple virtual machines running on a host through one shared physical network interface.
- Bonding allows failover redundancy for hosts with multiple physical network interfaces to the same network.
- VLANs can be used to separate network traffic logically while using only one (or more) physical network interface.

Configure bridging

The most common application scenario for *bridging* is the shared use of a physical network card by one or more virtual machines. Instead of one network card for each virtual machine and the virtualization server itself, all systems are connected via a shared uplink. A bridge can be compared with a switch implemented in software which is used to connect the individual hosts together. The hardware network adapter used is called a *bridge port*.

In order to configure a bridge, `Bridge` must be selected as the *Interface type* under *Add*. The *Name of new bridge interface* can be selected at will. Then click on *Next*.

The physical network card intended to act as the uplink can be selected under *Bridge ports*. In the typical scenario of connecting virtual machines via just one network card, there is no risk of a network loop. If the bridge is used to connect two Ethernet networks, the spanning tree protocol (STP) is employed to avoid network loops. The Linux kernel only implements STP, not the Rapid STP or Multiple STP versions.

The *Forwarding delay* setting configures the waiting time in seconds during which information is collected about the network topology when a connection is being made via STP. If the bridge is used for connecting virtual machines to one physical network card, STP should be disabled by setting the value to `0`. Otherwise problems may occur when using DHCP, as the packets sent during the waiting time are not forwarded.

The *Additional bridge options* input field can be used to configure arbitrary bridge parameters. This is only necessary in exceptional cases; an overview of the possible settings can be found on the manual page *bridge-utils-interfaces(5)*.

Clicking on *Next* offers the possibility of optionally assigning the bridge an IP address. This interface can then also be used as a network interface for the virtualization host. The options are the same as described in *Configuration of IPv4 addresses* (page 152) and *Configuration of IPv6 addresses* (page 152).

Configure bonding

Bonding can be used to bundle two (or more) physical network cards in order to increase the performance or improve redundancy in failover scenarios.

In order to configure a bonding, `Bonding` must be selected as the *Interface type* under *Add*. The *Name of the bonding interface* can be selected at will. Then click on *Next*.

The network cards which form part of the bonding interface are selected under *Bond slaves*. The network cards which should be given preference in failover scenarios (see below) can be selected via *Bond primary*.

The *Mode* configures the distribution of the network cards within the bonding:

- `balance-rr (0)` distributes the packets equally over the available network interfaces within the bonding one after the other. This increases performance and improves redundancy. In order to use this mode, the network switches used must support *link aggregation*.
- When `active-backup (1)` is used, only one network card is active for each bonding interface (by default this is the network interface configured in *Bond primary*). If the primary network card fails, this is detected by the Linux kernel, which switches to another card in the bonding. This version increases redundancy. It can be used with every network switch.

In addition, there are also a number of other bonding methods. These are generally only relevant for special cases and are described under Linux Ethernet Bonding Driver HOWTO[36].

The Media Independent Interface (MII) of the network cards is used to detect failed network adapters. The *MII link monitoring frequency* setting specifies the testing interval in milliseconds.

All other bonding parameters can be configured under *Additional bonding options*. This is only necessary in exceptional cases; an overview of the possible settings can be found under Linux Ethernet Bonding Driver HOWTO[37].

Clicking on *Next* allows to optionally assign the bonding interface an IP address. If one of the existing network cards which form part of the bonding interface has already been assigned an IP address, this configuration will be removed. The options are the same as described in *Configuration of IPv4 addresses* (page 152) and *Configuration of IPv6 addresses* (page 152).

Configure VLAN

VLANs can be used to separate the network traffic in a physical network logically over one or more virtual subnetworks. Each of these virtual networks is an independent broadcast domain. This makes it e.g. possible to differentiate between a network for the employees and a guest network for visitors in a company network although they use the same physical cables. The individual end devices can be assigned to the VLANs via the configuration of the switches. The network switches must support 802.1q VLANs.

A distinction is made between two types of connections between network cards:

- A connection only transports packets from a specific VLAN. In this case, untagged data packets are transmitted.

  This is typically the case if only one individual end device is connected via this network connection.

- A connection transports packets from several VLANs. This is also referred to as a trunk link. In this case, each packet is assigned to a VLAN using a VLAN ID. During transmission between trunk links and specific VLANs, the network switch takes over the task of filtering the packets by means of the VLAN IDs as well as adding and removing the VLAN IDs.

  This type of connection is primarily used between switches/servers.

  Some switches also allow the sending of packets with and without VLAN tags over a shared connection, but this is not described in more detail here.

When configuring a VLAN in the UMC module *Network settings* it is possible to configure for a computer which VLANs it wants to participate in. An example here would be an internal company web server, which should be available both to the employees and any users of the guest network.

[36] https://www.kernel.org/doc/Documentation/networking/bonding.txt
[37] https://www.kernel.org/doc/Documentation/networking/bonding.txt

In order to configure a VLAN, `Virtual LAN` must be selected as the *Interface type* under *Add*. The network interface for which the VLAN is specified with *Parent interface*. The *VLAN ID* is the unique identifier of the VLAN. Valid values are from 1 to 4095. Then *Next* must be clicked.

Clicking on *Next* allows to optionally assign the VLAN interface an IP address. The options are the same as described in *Configuration of IPv4 addresses* (page 152) and *Configuration of IPv6 addresses* (page 152). When assigning an IP address, ensure that the address matches the assigned VLAN address range.

8.2.5 Proxy access configuration

The majority of the command line tools which access web servers (e.g., **`wget`**, **`elinks`** or **`curl`**) check whether the environment variables `http_proxy` or `https_proxy` are set. If this is the case, the proxy server set in these variables is used automatically.

The Univention Configuration Registry Variable *`proxy/http`* (page 303) and *`proxy/https`* (page 303) can also be used to activate the setting of these environment variables through an entry in `/etc/profile`.

The proxy URL must be specified for this, for example `http://192.0.2.100`. The proxy port can be specified in the proxy URL using a colon, for example `http://192.0.2.100:3128`. If the proxy requires authentication, this can be provided in the form `http://`*`username`*`:`*`password`*`@192.0.2.100`.

The environment variable is not adopted for sessions currently opened. A new login is required for the change to be activated.

The Univention tools for software updates also support operation via a proxy and query the Univention Configuration Registry variable.

Individual domains can be excluded from use by the proxy by including them separated by commas in the Univention Configuration Registry Variable *`proxy/no_proxy`* (page 303). Subdomains are taken into account; e.g. an exception for `software-univention.de` also applies for `updates.software-univention.de`.

8.2.6 Mounting NFS shares

The *NFS mounts* policy of the UMC computer management can be used to configure NFS shares, which are mounted on the system. There is a *NFS share* for selection, which is mounted in the file path specified under *Mount point*.

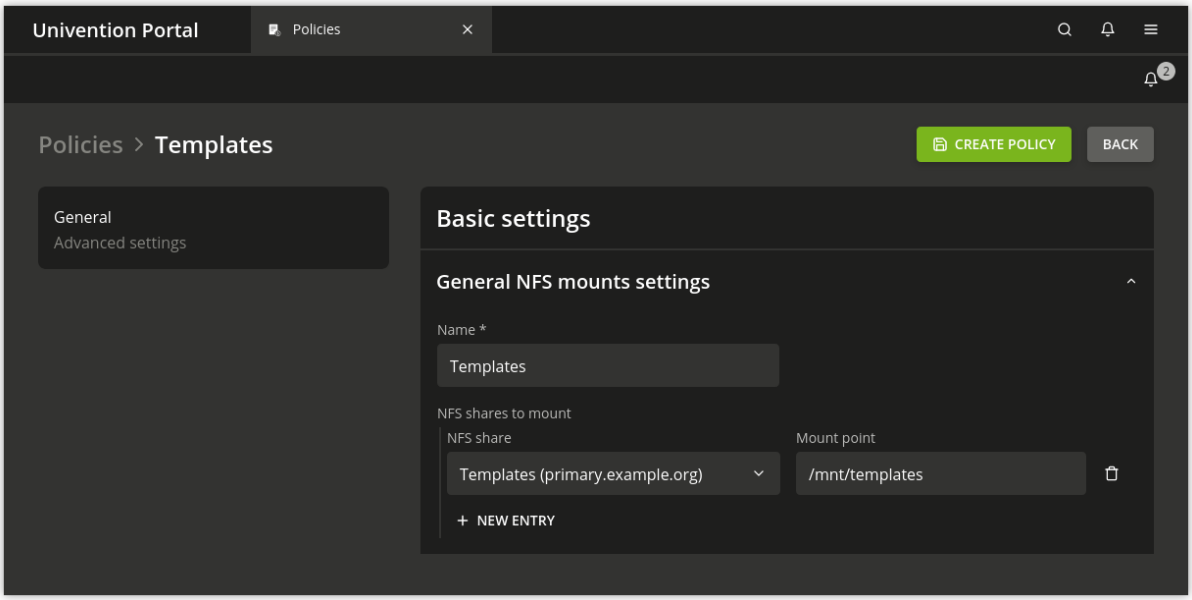


Fig. 8.5: Mounting a NFS share

8.2.7 Collection of list of supported hardware

Univention collects information about hardware which is compatible with UCS and in use by customers. The information processed for this is gathered by the UMC module *Hardware information*.

All files are forwarded to Univention anonymously and only transferred once permission has been received from the user.

The start dialogue contains the entry fields *Manufacturer* and *Model*, which must be completed with the values determined from the DMI information of the hardware. The fields can also be adapted and an additional *Descriptive comment* added.

If the hardware information is transferred as part of a support request, the *This is related to a support case* option should be activated. A ticket number can be entered in the next field; this facilitates assignment and allows quicker processing.

Clicking on *Next* offers an overview of the transferred hardware information. In addition, a compressed TAR archive is created, which contains a list of the hardware components used in the system and can be downloaded via *Archive with system information*.

Clicking on *Next* again allows you to select the way the data are transferred to Univention. *Upload* transmits the data via HTTPS, *Send mail)* opens a dialogue, which lists the needed steps to send the archive via email.

8.3 Administration of local system configuration with Univention Configuration Registry

Univention Configuration Registry is the central tool for managing the local system configuration of a UCS-based system. Direct editing of the configuration files is usually not necessary.

Settings are specified in a consistent format in a registry mechanism, the so-called *Univention Configuration Registry variables*. These variables are used to generate the configuration files used effectively by the services/programs from the configuration templates (the so-called *Univention Configuration Registry templates*).

This procedure offers a range of advantages:

- It is not usually necessary to edit any configuration files manually. This avoids errors arising from invalid syntax of configuration settings or similar.
- There is a uniform interface for editing the settings and the different syntax formats of the configuration files are hidden from the administrator.
- Settings are decoupled from the actual configuration file, i.e., if a software uses a different configuration format in a new version, a new template in a new format is simply delivered instead of performing time-consuming and error-prone conversion of the file.
- The variables used in a configuration file administrated with Univention Configuration Registry are registered internally. This ensures that when a UCR variable is changed, all the configuration files containing the changed variable are recreated.

Univention Configuration Registry variables can be configured in the command line using the **univention-config-registry** command (short form: **ucr**) or via the UMC module *Univention Configuration Registry*.

As the majority of packages perform their configuration via Univention Configuration Registry and the corresponding basic settings need to be set up during the installation, hundreds of Univention Configuration Registry variables are already set after the installation of a UCS system.

UCR variables can also be used efficiently in shell scripts for accessing current system settings.

The variables are named according to a tree structure with a forward slash being used to separate components of the name. For example, Univention Configuration Registry variables beginning with `ldap` are settings which apply to the local directory service.

A description is given for the majority of variables explaining their use.

If a configuration file is administrated by a UCR template and the required setting has not already been covered by an existing variable, the UCR template should be edited instead of the configuration file. If the configuration were directly adapted, the next time the file is regenerated - e.g., when a registered UCR variable is set - the local modification will be overwritten again. Adaptation of UCR templates is described in *Modifying UCR templates* (page 160).

Part of the settings configured in Univention Configuration Registry are system-specific (e.g., the computer name); many settings can, however, be used on more then one computer. The Univention Configuration Registry policy in the domain administration UMC modules can be used to compile variables and apply them on more than one computer.

The evaluation of the Univention Configuration Registry variables on a UCS system comprises four stages:

- First the local Univention Configuration Registry variables are evaluated.
- The local variables are overruled by policy variables which are usually sourced from the directory service
- The `--schedule` option is used to set local variables which are only intended to apply for a certain period of time. This level of the Univention Configuration Registry is reserved for local settings which are automated by time-controlled mechanisms in Univention Corporate Server.
- When the `--force` option is used in setting a local variable, settings adopted from the directory service and variables from the schedule level are overruled and the given value for the local system fixed instead. An example:

```
$ univention-config-registry set --force mail/messagesizelimit=1000000
```

If a variable is set which is overwritten by a superordinate policy, a warning message is given.

The use of the Univention Configuration Registry policy is documented in the *Policy-based configuration of UCR variables* (page 160).

8.3.1 Using the Univention Management Console module

The UMC module *Univention Configuration Registry* can be used to display and adjust the variables of a system. There is also the possibility of setting new variables using *Add new variable*.

A search mask is displayed on the start page. All variables are classified using a *Category*, for example all LDAP-specific settings.

The *Search attribute* can be entered as a filter in the search mask, which can refer to the variable name, value or description.

Following a successful search, the variables found are displayed in a table with the variable name and the value. A detailed description of the variable is displayed when moving the mouse cursor over the variable name.

A variable can be edited by clicking on its name. A variable can be deleted by right-clicking and selecting *Delete*.

8.3.2 Using the command line frontend

The command line interface of Univention Configuration Registry is run using the **univention-config-registry** command. Alternatively, the short form **ucr** can be used.

Querying a UCR variable

get

A single Univention Configuration Registry variable can be queried with the parameter *get* (page 158):

```
$ univention-config-registry get ldap/server/ip
```

dump

The parameter *dump* (page 158) can also be used to display all currently set variables:

```
$ univention-config-registry dump
```

Setting UCR variables

set

The parameter *set* (page 158) is used to set a variable. The variable can be given any name consisting exclusively of letters, full stops, figures, hyphens and forward slashes.

```
$ univention-config-registry set VARIABLENAME=VALUE
```

If the variable already exists, the content is updated; otherwise, a new entry is created.

When setting a new value for a Univention Configuration Registry variable UCR runs checks to verify the compatibility of the value with the variable type. In case of incompatibility UCR displays a warning message. Moreover, the variable is not set to the new value if the Univention Configuration Registry variable *ucr/check/type* (page 306) is `true` (default is `false`). If the `--ignore-check` option is used, the value is always set independent of type compatibility and setting of *ucr/check/type* (page 306).

When a variable changes, UCR rewrites all configuration files immediately for which the variable is registered. UCR outputs the paths of the updated files to the console.

In doing so it must be noted that although the configuration of a service is updated, the service in question is not restarted automatically! The restart must be performed manually.

It is also possible to perform simultaneous changes to several variables in one command line. If these refer to the same configuration file, the file is only rewritten once.

```
$ univention-config-registry set \
  dns/forwarder1=192.0.2.2 \
  sshd/xforwarding="no" \
  sshd/port=2222
```

A conditional setting is also possible. For example, if a value should only be saved in a Univention Configuration Registry variable when the variable does not yet exist, this can be done by entering a question mark (`?`) instead of the equals sign (=) when assigning values.

```
$ univention-config-registry set dns/forwarder1?192.0.2.2
```

Searching for variables and set values

search

The parameter *search* (page 158) can be used to search for a variable. This command searches for variable names which contain `nscd` and displays these with their current assignments:

```
$ univention-config-registry search nscd
```

Alternatively, searches can also be performed for set variable values. This request searches for all variables set to `primary.example.com`:

```
$ univention-config-registry search --value primary.example.com
```

Search templates in the form of regular expressions can also be used in the search. The complete format is documented at Regular expression operations in the Python 3 documentation[38].

Deleting UCR variables

unset

The parameter *unset* (page 159) is used to delete a variable. The following example deletes the variable *dns/forwarder2* (page 293). It is also possible here to specify several variables to be deleted:

```
$ univention-config-registry unset dns/forwarder2
```

Regeneration of configuration files from their template

commit

The parameter *commit* (page 159) is used to regenerate a configuration file from its template. The name of the configuration file is entered as a parameter, e.g.:

```
$ univention-config-registry commit /etc/samba/smb.conf
```

As UCR templates are generally regenerated automatically when UCR variables are edited, this is primarily used for tests.

If no filename is given when running **ucr commit**, all of the files managed by Univention Configuration Registry will be regenerated from the templates. It is, however, not generally necessary to regenerate all the configuration files.

Sourcing variables in shell scripts

shell

The parameter *shell* (page 159) is used to display Univention Configuration Registry variables and their current assignments in a format that can be used in shell scripts.

```
$ univention-config-registry shell ldap/server/name
```

Different conversions are involved in this: forward slashes in variable names are replaced with underscores and characters in the values which have a particular significance in shell scripts are included in quotation marks to ensure they are not altered.

The Univention Configuration Registry output must be executed via the command **eval** for Univention Configuration Registry variables to be able to be read in a shell script as environment variables:

```
# eval "$(univention-config-registry shell ldap/server/name)"
# echo "$ldap_server_name"
primary.firma.de
```

[38] https://docs.python.org/3/library/re.html

8.3.3 Policy-based configuration of UCR variables

Part of the settings configured in Univention Configuration Registry are system-specific (e.g., the computer name); many settings can, however, be used on more then one computer. The *Univention Configuration Registry* policy managed in the UMC module *Policies* can be used to compile variables and apply them on more than one computer.

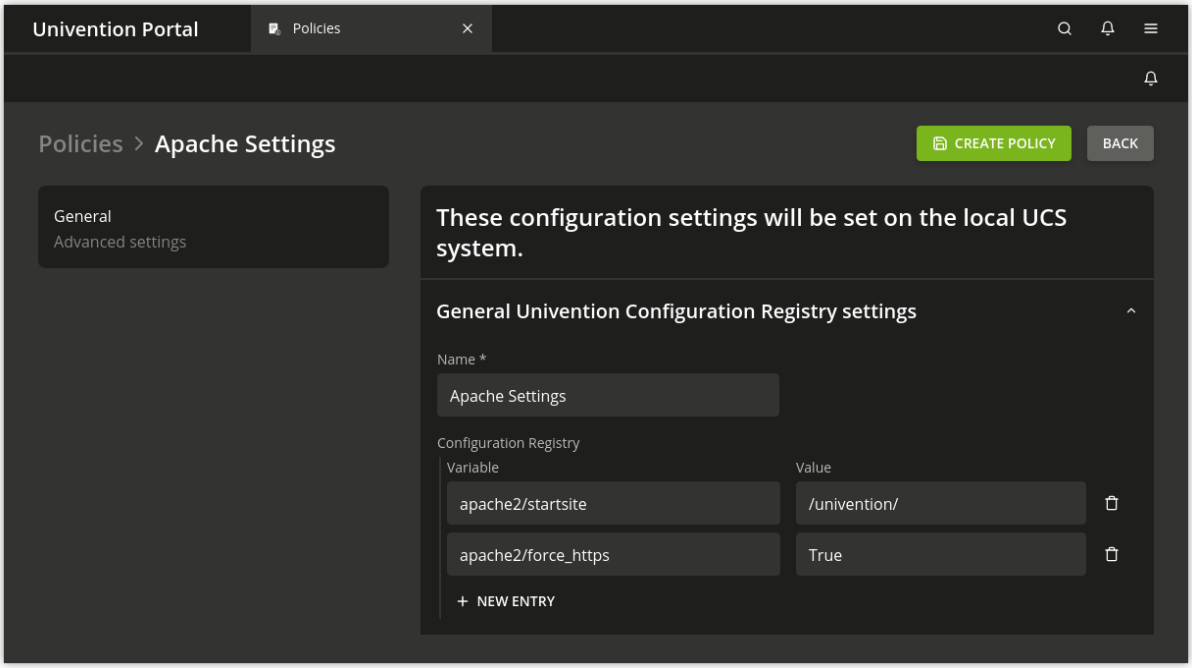


Fig. 8.6: Policy-based configuration of the Apache start page and forced HTTPS

Firstly, a *Name* must be set for the policy which is to be created, under which the variables will later be assigned to the individual computer objects.

In addition, at least one *Variable* must be configured and a *Value* assigned.

This policy can then be assigned to a computer object or a container or OU (see *Applying policies* (page 75)). Note that the evaluation of configured values differs from other policies: The values are not forwarded directly to the computer, but rather written on the assigned computer by Univention Directory Policy. The time interval used for this is configured by the Univention Configuration Registry Variable `ldap/policy/cron` (page 296) and is set to hourly as standard.

8.3.4 Modifying UCR templates

In the simplest case, a Univention Configuration Registry template is a copy of the original configuration file in which the points at which the value of a variable are to be used contain a reference to the variable name.

Inline Python code can also be integrated for more complicated scenarios, which then also allows more complicated constructions such as conditional assignments.

Note: Univention Configuration Registry templates are included in the corresponding software packages as configuration files. When packages are updated, a check is performed for whether any changes have been made to the configuration files.

If configuration files are no longer there in the form in which they were delivered, they will not be overwritten. Instead a new version will be created in the same directory with the ending `.debian.dpkg-new`.

If changes are to be made on the Univention Configuration Registry templates, these templates are also not overwritten during the update and are instead re-saved in the same directory with the ending `.dpkg-new` or `.dpkg-dist`.

Corresponding notes are written in the `/var/log/univention/actualise.log` log file. This only occurs if UCR templates have been locally modified.

The UCR templates are stored in the `/etc/univention/templates/files/` directory. The path to the templates is the absolute path to the configuration file with the prefixed path to the template directory. For example, the template for the `/etc/issue` configuration file can be found under `/etc/univention/templates/files/etc/issue`.

For the configuration files to be processed correctly by Univention Configuration Registry they must be in UNIX format. If configuration files are edited in DOS or Windows, for example, control characters are inserted to indicate line breaks, which can disrupt the way Univention Configuration Registry uses the file.

Referencing of UCR variables in templates

In the simplest case, a UCR variable can be directly referenced in the template. The variable name framed by the string `@%@` represents the wildcard. As an example the option for the activation of X11 forwarding in the configuration file `/etc/ssh/sshd_config` of the OpenSSH server:

```
X11Forwarding @%@sshd/xforwarding@%@
```

Newly added references to UCR variables are automatically evaluated by templates; additional registration is only required with the use of inline Python code (see *Integration of inline Python code in templates* (page 161)).

Integration of inline Python code in templates

Any type of Python code can be embedded in UCR templates by entering a code block framed by the string `@!@`. For example, these blocks can be used to realize conditional requests so that when a parameter is changed via a variable, further dependent settings are automatically adopted in the configuration file. The following code sequence configures for example network settings using the Univention Configuration Registry settings:

```
@!@
if configRegistry.get('apache2/ssl/certificate'):
    print('SSLCertificateFile %s' %
        configRegistry['apache2/ssl/certificate'])
@!@
```

All the data output with the print function are written in the generated configuration file. The data saved in Univention Configuration Registry can be requested via the `ConfigRegistry` object, e.g.:

```
@!@
if configRegistry.get('version/version') and \
        configRegistry.get('version/patchlevel'):
    print('UCS %(version/version)s-%(version/patchlevel)s' %
        configRegistry)
@!@
```

In contrast to directly referenced UCR variables (see *Referencing of UCR variables in templates* (page 161)), variables accessed in inline Python code must be explicitly registered.

The Univention Configuration Registry variables used in the configuration files are registered in *info* files in the `/etc/univention/templates/info/` directory which are usually named after the package name with the file ending `.info`. If new Python code is entered into the templates or the existing code changed in such a way that it requires additional or different variables, one of the existing `.info` files will need to be modified or a new one added.

Following the changing of `.info` files, the **`ucr update`** command must be run.

8.4 Basic system services

This chapter describes basic system services of a UCS Installation such as the configuration of the PAM authentication framework, system logs and the NSCD.

8.4.1 Administrative access with the root account

There is a `root` account on every UCS system for complete administrative access. The password is set during installation of the system. The root user **is not** stored in the LDAP directory, but instead in the local user accounts.

The password for the root user can be changed via the command line by using the **passwd** command. It must be pointed out that this process does not include any checks regarding either the length of the password or the passwords used in the past.

8.4.2 Configuration of language and keyboard settings

In Linux, localization properties for software are defined in so-called *locales*. Configuration includes, among other things, settings for date and currency format, the set of characters in use and the language used for internationalized programs. The installed locales can be changed in the UMC module *Language settings* under *Language settings* ‣ *Installed system locales*. The standard locale is set under *Default system locale*.

The *Keyboard layout* in the menu entry *Time zone and keyboard settings* is applied during local logins to the system.

8.4.3 Starting/stopping system services / configuration of automatic startup

The UMC module *System services* can be used to check the current status of a system service and to start or stop it as required.

In this list of all the services installed on the system, the current running runtime status and a *Description* are displayed under *Status*. The service can be started, stopped or restarted under *more*.

By default every service is started automatically when the system is started. In some situations, it can be useful not to have the service start directly, but instead only after further configuration. The action *Start manually* is used so that the service is not started automatically when the system is started, but can still be started subsequently. The action *Start never* also prevents subsequent service starts.

8.4.4 Authentication / PAM

Authentication services in Univention Corporate Server are realized via *Pluggable Authentication Modules* (PAM). To this end different login procedures are displayed on a common interface so that a new login method does not require adaptation for existing applications.

Limiting authentication to selected users

By default only the `root` user and members of the `Domain Admins` group can login remotely via SSH and locally on a `tty`.

This restriction can be configured with the Univention Configuration Registry Variable `auth/`*`SERVICE`*`/restrict`. Access to this service can be authorized by setting the variables `auth/`*`SERVICE`*`/user/`*`USERNAME* and `auth/`*`SERVICE`*`/group/`*`GROUPNAME* to `yes`.

Login restrictions are supported for *SSH* (`sshd`), login on a *tty* (`login`), *rlogin* (`rlogin`), *PPP* (`ppp`) and other services (`other`). An example for *SSH*:

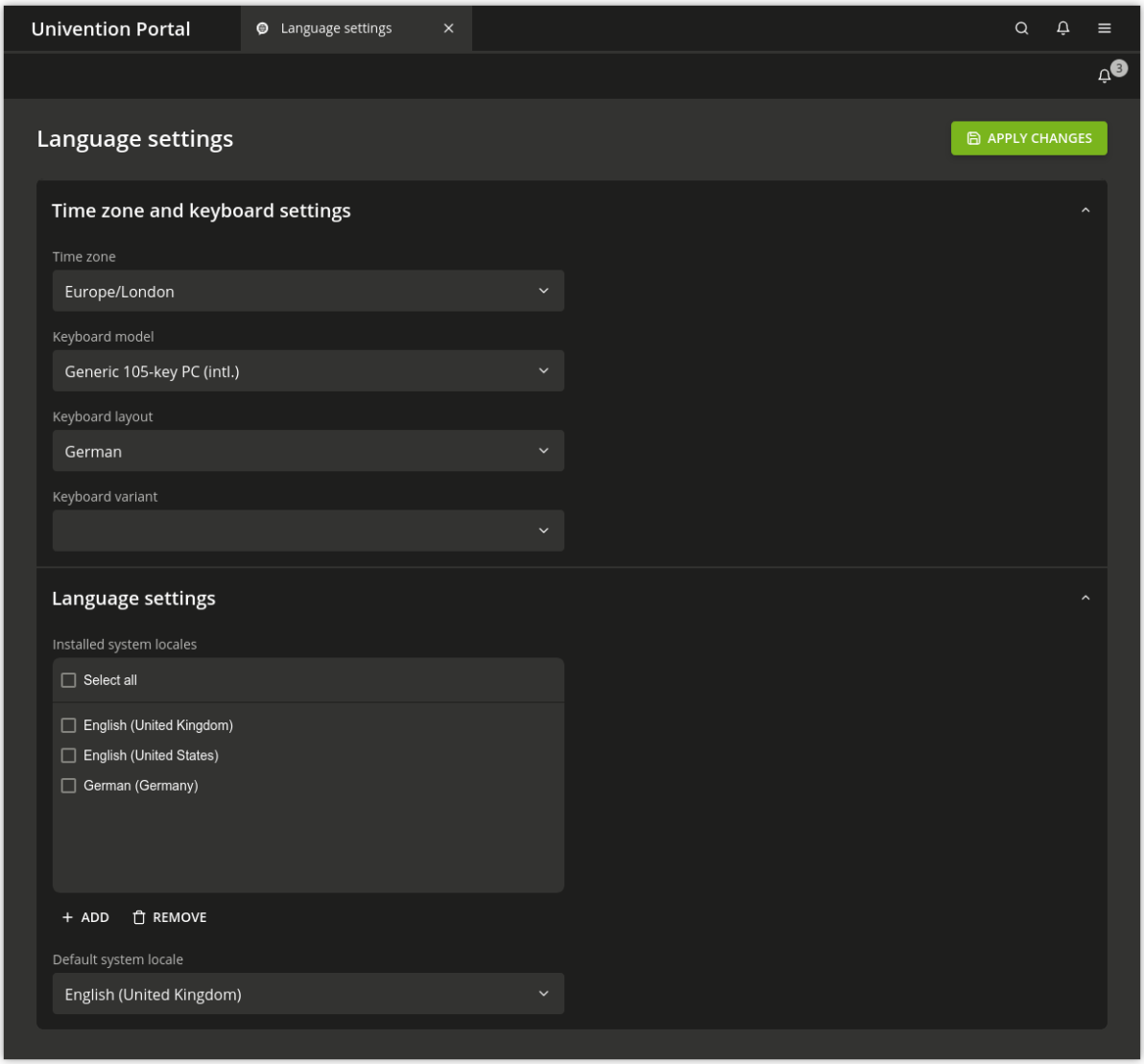


Fig. 8.7: Configuring the language settings

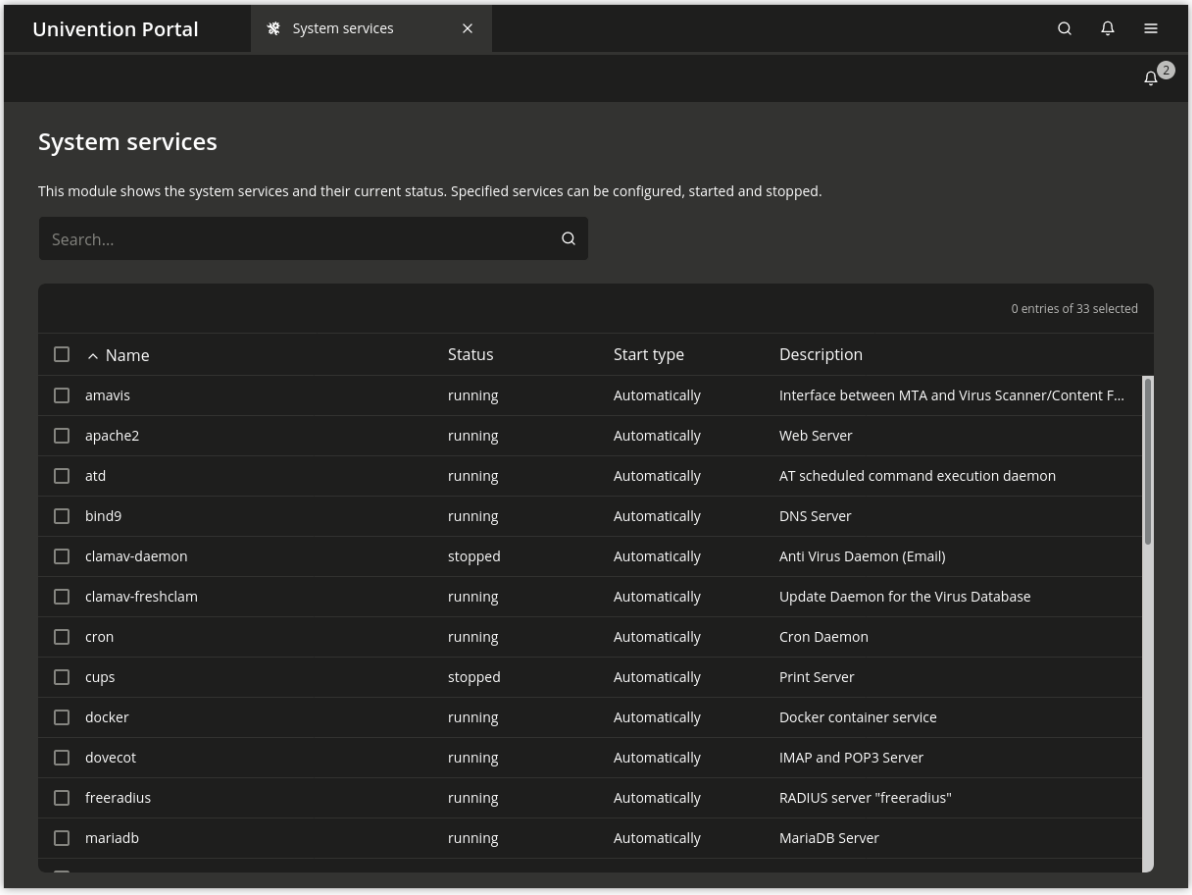


Fig. 8.8: Overview of system services

```
auth/sshd/group/Administrators: yes
auth/sshd/group/Computers: yes
auth/sshd/group/DC Backup Hosts: yes
auth/sshd/group/DC Slave Hosts: yes
auth/sshd/group/Domain Admins: yes
auth/sshd/restrict: yes
```

8.4.5 Configuration of the LDAP server in use

Several LDAP servers can be operated in a UCS domain. The primary one used is specified with the Univention Configuration Registry Variable *ldap/server/name* (page 297), further servers can be specified via the Univention Configuration Registry Variable *ldap/server/addition* (page 297).

Alternatively, the LDAP servers can also be specified via a *LDAP server* policy. The order of the servers determines the order of the computer's requests to the server if a LDAP server cannot be reached.

By default only *ldap/server/name* (page 297) is set following the installation or the domain join. If there is more than one LDAP server available, it is advisable to assign at least two LDAP servers using the *LDAP server* policy in order to improve redundancy. In cases of an environment distributed over several locations, preference should be given to LDAP servers from the local network.

8.4.6 Configuration of the print server in use

The print server to be used can be specified with the Univention Configuration Registry Variable *cups/server* (page 292).

Alternatively, the server can also be specified via the *Print server* policy in the UMC module *Computers*.

8.4.7 Logging/retrieval of system messages and system status

Log files

All UCS-specific log files (e.g., for the listener/notifier replication) are stored in the `/var/log/univention/` directory. Services write log messages their own standard log files: for example, Apache to the file `/var/log/apache2/error.log`.

The log files are managed by **logrotate**. It ensures that log files are named in series in intervals (can be configured in weeks using the Univention Configuration Registry Variable *log/rotate/weeks* (page 297), with the default setting being 12) and older log files are then deleted. For example, the current log file for the Univention Directory Listener is found in the `listener.log` file; the one for the previous week in `listener.log.1`, etc.

Alternatively, log files can also be rotated only once they have reached a certain size. For example, if they are only to be rotated once they reach a size of 50 MB, the Univention Configuration Registry Variable *logrotate/rotates* (page 297) can be set to `size 50M`.

The Univention Configuration Registry Variable *logrotate/compress* (page 297) is used to configure whether the older log files are additionally zipped with **gzip**.

Log files located in the directory `/var/log/univention/listener_modules` each have their own Logrotate configuration. These log files have global and specific Logrotate settings. The Univention Configuration Registry Variable `logrotate/listener-modules/<directive>` configures the global settings. The *logrotate(8)* documentation describes the functionality in detail. UCS supports the following directives:

logrotate/listener-modules/rotate

Default value: `weekly`

logrotate/listener-modules/rotate/count

Default value: `12`

logrotate/listener-modules/create
Default value: `640 listener adm`

logrotate/listener-modules/missingok
Default value: `missingok`

logrotate/listener-modules/compress
Default value: `compress`

logrotate/listener-modules/notifempty
Default value: `notifempty`

If a configuration only applies to a specific log file, compose the Univention Configuration Registry Variable as follows: `logrotate/listener-modules/<logfile-name>/<directive>`. Use the log filename without the file suffix `.log`.

Logging the system status

univention-system-stats can be used to document the current system status in the `/var/log/univention/system-stats.log` file. The following values are logged:

- The free disk space on the system partitions (**df -lhT**)
- The current process list (**ps auxf**)
- Two **top** lists of the current processes and system load (**top -b -n2**)
- The current free system memory (**free**)
- The time elapsed since the system was started (**uptime**)
- Temperature, fan and voltage indexes from **lm-sensors** (**sensors**)
- A list of the current Samba connections (**smbstatus**)

The runtime in which the system status should be logged can be defined in Cron syntax via the Univention Configuration Registry Variable *system/stats/cron* (page 305), e.g., `0,30 * * * *` for logging every half and full hour. The logging is activated by setting the Univention Configuration Registry Variable *system/stats* (page 305) to `yes`. This is the default since UCS 3.0.

Process overview via Univention Management Console module

The UMC module *Process overview* displays a table of the current processes on the system. The processes can be sorted based on the following properties by clicking on the corresponding table header:

- CPU utilization in percent
- The username under which the process is running
- Memory consumption in percent
- The process ID

The menu item *more* can be used to terminate processes. Two different types of termination are possible:

Terminate
The action *Terminate* sends the process a `SIGTERM` signal; this is the standard method for the controlled termination of programs.

Force terminate
Sometimes, it may be the case that a program - e.g., after crashing - can no longer be terminated with this procedure. In this case, the action *Force terminate* can be used to send the signal `SIGKILL` and force the process to terminate.

As a general rule, terminating the program with `SIGTERM` is preferable as many programs then stop the program in a controlled manner and, for example, save open files.

System diagnostic via Univention Management Console module

The UMC module *System diagnostic* offers a corresponding user interface to analyze a UCS system for a range of known problems.

The module evaluates a range of problem scenarios known to it and suggests solutions if it is able to resolve the identified solutions automatically. This function is displayed via ancillary buttons. In addition, links are shown to further articles and corresponding UMC modules.

8.4.8 Executing recurring actions with Cron

Regularly recurring actions (e.g., the processing of log files) can be started at a defined time with the Cron service. Such an action is known as a cron job.

Hourly/daily/weekly/monthly execution of scripts

Four directories are predefined on every UCS system, `/etc/cron.hourly/`, `/etc/cron.daily/`, `/etc/cron.weekly/` and `/etc/cron.monthly/`. Shell scripts which are placed in these directories and marked as executable are run automatically every hour, day, week or month.

Defining local cron jobs in `/etc/cron.d/`

A cron job is defined in a line, which is composed of a total of seven columns:

- Minute (0-59)
- Hour (0-23)
- Day (1-31)
- Month (1-12)
- Weekday (0-7) (0 and 7 both stand for Sunday)
- Name of user executing the job (e.g., `root`)
- The command to be run

The time specifications can be set in different ways. One can specify a specific minute/hour/etc. or run an action every minute/hour/etc. with a `*`. Intervals can also be defined, for example `*/2` as a minute specification runs an action every two minutes.

Example:

```
30 * * * * root /usr/sbin/jitter 600 /usr/share/univention-samba/slave-sync
```

Defining cron jobs in Univention Configuration Registry

Cron jobs can also be defined in Univention Configuration Registry. This is particularly useful if they are set via a Univention Directory Manager policy and are thus used on more than one computer.

Each cron job is composed of at least two Univention Configuration Registry variables. *JOBNAME* is a general description.

- `cron/`*JOBNAME*`/command` specifies the command to be run (required)
- `cron/`*JOBNAME*`/time` specifies the execution time (see *Defining local cron jobs in /etc/cron.d/* (page 167)) (required)
- As standard, the cron job is run as a user `root`. `cron/`*JOBNAME*`/user` can be used to specify a different user.

- If an email address is specified under `cron/JOBNAME/mailto`, the output of the cron job is sent there per email.
- `cron/JOBNAME/description` can be used to provide a description.

8.4.9 Name service cache daemon

Data of the NSS service is cached by the *Name Server Cache Daemon* (NSCD) in order to speed up frequently recurring requests for unchanged data. Thus, if a repeated request occurs, instead of a complete LDAP request to be processed, the data are simply drawn directly from the cache.

Since UCS 3.1, the groups are no longer cached via the NSCD for performance and stability reasons; instead they are now cached by a local group cache, see *Local group cache* (page 139).

The central configuration file of the (`/etc/nscd.conf`) is managed by Univention Configuration Registry.

The access to the cache is handled via a hash table. The size of the hash table can be specified in Univention Configuration Registry, and should be higher than the number of simultaneously used users/hosts. For technical reasons, a prime number should be used for the size of the table. The following table shows the standard values of the variables:

Table 8.8: Default size of the hash table

Variable	Default size of the hash table
`nscd/hosts/size`	`6007`
`nscd/passwd/size`	`6007`

With very big caches it may be necessary to increase the size of the cache database in the system memory. This can be configured through the Univention Configuration Registry variables *nscd/hosts/maxdbsize* (page 300), *nscd/group/maxdbsize* (page 300) and *nscd/passwd/maxdbsize* (page 300).

As standard, five threads are started by NSCD. In environments with many accesses it may prove necessary to increase the number via the Univention Configuration Registry Variable *nscd/threads* (page 300).

In the basic setting, a resolved group or hostname is kept in cache for one hour, a username for ten minutes. With the Univention Configuration Registry variables *nscd/group/positive_time_to_live* (page 300), *nscd/hosts/positive_time_to_live* (page 300) and *nscd/passwd/positive_time_to_live* (page 300) these periods can be extended or diminished (in seconds).

From time to time it might be necessary to manually invalidate the cache of the NSCD. This can be done individually for each cache table with the following commands:

```
$ nscd -i passwd
$ nscd -i hosts
```

The verbosity of the log messages can be configured through the Univention Configuration Registry Variable *nscd/debug/level* (page 300).

8.4.10 SSH login to systems

When installing a UCS system, an SSH server is also installed per preselection. SSH is used for realizing encrypted connections to other hosts, wherein the identity of a host can be assured via a check sum. Essential aspects of the SSH server's configuration can be adjusted in Univention Configuration Registry.

By default the login of the privileged `root` user is permitted by SSH (e.g. for configuring a newly installed system where no users have been created yet, from a remote location).

- If the Univention Configuration Registry Variable *sshd/permitroot* (page 305) is set to `without-password`, then no interactive password request will be performed for the `root` user, but only a login based on a public key. By this means brute force attacks to passwords can be avoided.

- To prohibit SSH login completely, this can be deactivated by setting the Univention Configuration Registry Variable *auth/sshd/user/root* (page 291) to `no`.

The Univention Configuration Registry Variable *sshd/xforwarding* (page 305) can be used to configure whether an X11 output should be passed on via SSH. This is necessary, for example, for allowing a user to start a program with graphic output on a remote computer by logging in with **`ssh -X TARGETHOST`**. Valid settings are `yes` and `no`.

The standard port for SSH connections is port 22 via TCP. If a different port is to be used, this can be arranged via the Univention Configuration Registry Variable *sshd/port* (page 305).

8.4.11 Configuring the time zone / time synchronization

The time zone in which a system is located can be changed in the UMC module *Language settings* under *Time zone and keyboard settings* ‣ *Time zone*.

Asynchronous system times between individual hosts of a domain can be the source of a large number of errors, for example:

- The reliability of log files is impaired.
- Kerberos operation is disrupted.
- The correct evaluation of the validity periods of passwords can be disturbed.

Usually the Primary Directory Node functions as the time server of a domain. With the Univention Configuration Registry variables *timeserver* (page 306), *timeserver2* (page 306) and *timeserver3* (page 306) external NTP servers can be included as time sources.

Manual time synchronization can be started by the command **`ntpdate`**.

Windows clients joined in a Samba/AD domain only accept signed NTP time requests. If the Univention Configuration Registry Variable *ntp/signed* (page 301) is set to `yes`, the NTP replies are signed by Samba/AD.

CHAPTER

NINE

SERVICES FOR WINDOWS

UCS can offer Active Directory (AD) services, be a member of an Active Directory domain or synchronize objects between Active Directory domains and a UCS domain.

For the purposes of Windows systems, UCS can assume the tasks of Windows server systems:

- Domain controller function / authentication services
- File services
- Print services

In UCS all these services are provided by Samba.

UCS supports the mostly automatic migration of an existing Active Directory domain to UCS. All users, groups, computer objects and group policies are migrated without the need to rejoin the Windows clients. This is documented in *Migrating an Active Directory domain to UCS using Univention AD Takeover* (page 196).

Microsoft Active Directory domain controllers cannot join the Samba domain. This functionality is planned at a later point in time.

Samba can not join an Active Directory Forest yet at this point.

Incoming trust relationships with other Active Directory domains are configurable. In this setup the external Active Directory domain trusts authentication decisions of the UCS domain (Windows trusts UCS) so that UCS users can sign in to systems and Active Directory backed services in the Windows domain (see *Trust relationships* (page 200)). Outgoing trusts with Active Directory domain (UCS trusts Windows) are not supported currently.

9.1 Operation of a Samba domain based on Active Directory

9.1.1 Installation

Samba as an AD domain controller can be installed on all UCS Directory Nodes from the Univention App Center with the application **Active Directory-compatible domain controller**. Alternatively, the software package **univention-samba4** can be installed. On the system roles Primary Directory Node and Backup Directory Node the **univention-s4-connector** package must also be installed and **univention-run-join-scripts** command must be run after installation. Additional information can be found in *Installation of further software* (page 99).

A Samba member server can be installed on UCS Managed Nodes from the Univention App Center with the application **Windows-compatible Fileserver**. Alternatively, the software package **univention-samba** can be installed (**univention-run-join-scripts** command must be run after installation). Additional information can be found in *Installation of further software* (page 99).

Samba supports the operation as a *read-only domain controller*. The setup is documented in *Extended Windows integration documentation* [7].

9.1.2 Services of a Samba domain

Authentication services

User logins can only be performed on Microsoft Windows systems joined in the Samba domain. Domain joins are documented in *Windows domain joins* (page 31).

Users who login to a Windows system are supplied with a Kerberos ticket when they login. The ticket is then used for the further authentication. This ticket allows access to the domain's resources.

Common sources of error in failed logins are:

- Synchronization of the system times between the Windows client and domain controller is essential for functioning Kerberos authentication. By default the system time is updated via NTP during system startup. This can also be done manually using the command **w32tm /resync** on the Windows client.
- DNS service records need to be resolved during login. For this reason, the Windows client should use the domain controller's IP address as its DNS name server.

File services

A file server provides files over the network and allows concentrating the storage of user data on a central server.

The file services integrated in UCS support the provision of shares using the CIFS protocol (see *File share management* (page 237)). Insofar as the underlying file system supports Access Control Lists (ACLs) (can be used with `ext4` and `XFS`), the ACLs can also be used by Windows clients.

Samba Active Directory domain controllers, i.e. UCS Directory Nodes, can also provide file services. As a general rule, it is recommended to separate domain controllers and file/print services in Samba environments - following the Microsoft recommendations for Active Directory - that means using UCS Directory Nodes for logins/authentication and UCS Managed Nodes for file/print services. This ensures that a high system load on a file server does not result in disruptions to the authentication service. For smaller environments in which it is not possible to run two servers, file and print services can also be run on a UCS Directory Node.

Samba supports the *CIFS* protocol and the successor *SMB2* to provide file services. Using a client which supports *SMB2* (as of **Windows Vista**, i.e., **Windows 7/8** too) improves the performance and scalability.

The protocol can be configured using the Univention Configuration Registry variable *samba/max/protocol* (page 303). It must be set on all Samba servers and then all Samba server(s) restarted.

- `NT1` configures *CIFS* (supported by all Windows versions)
- `SMB2` *SMB2* (supported as of **Windows Vista** / **Windows 7**)
- `SMB3` configures *SMB3* (supported as of **Windows 8**)

Print services

Samba offers the possibility of sharing printers set up under Linux as network printers for Windows clients. The management of the printer shares and the provision of the printer drivers is described in *Print services* (page 249).

Samba AD domain controllers can also provide print services. In this case, the restrictions described in *File services* (page 172) must be taken into consideration.

Univention S4 connector

When using Samba as an Active Directory domain controller, Samba provides a separate LDAP directory service. The synchronization between the UCS LDAP and the Samba LDAP occurs via an internal system service, the *Univention S4 connector*. The connector is enabled on the Primary Directory Node by default and typically requires no further configuration.

Further information on the status of the synchronization can be found in the log file `/var/log/univention/connector-s4.log`. Additional information on analyzing connector replication problems can be found in KB 32 - Samba 4 Troubleshooting[39].

The **`univention-s4search`** command can be used to search in the Samba directory service. If it is run as the `root` user, the required credentials of the machine account are used automatically:

```
$ root@primary:~# univention-s4search sAMAccountName=Administrator
# record 1
dn: CN=Administrator,CN=Users,DC=example,DC=com
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: user
cn: Administrator
instanceType: 4
(..)
```

Replication of directory data

Samba/AD domains use the Directory Replication System (DRS) to replicate the directory data. DRS allows multi-master replication, i.e., the write changes from multiple domain controllers are synchronized at protocol level. Consequently, the use of snapshots in virtualization solutions should be avoided when using Samba/AD and Samba/AD should be operated on a server which is never switched off.

The complexity of the multi-master replication increases with each additional Samba/AD domain controller. Consequently, it must be checked whether additional Samba/AD domain controllers provided by UCS Directory Nodes are necessary or if a UCS Managed Node would not be a better choice for new servers.

Additional information on troubleshooting replication problems can be found in KB 32 - Samba 4 Troubleshooting[40].

Synchronization of the SYSVOL share

The SYSVOL share is a share which provides group policies and logon scripts in Active Directory/Samba. It is synchronized among all domain controllers and stored in the `/var/lib/samba/sysvol/` directory.

In Microsoft Active Directory, the SYSVOL share is synchronized by the File Replication Service (introduced with **`Windows 2000`**) or the Distributed File System (as of **`Windows 2008 R2`**). These replication methods are not yet fully implemented in Samba/AD. The synchronization between the Samba/AD domain controllers is performed in UCS via a Cron job (every five minutes as standard - can be configured using the Univention Configuration Registry Variable *samba4/sysvol/sync/cron* (page 304)).

[39] https://help.univention.com/t/32
[40] https://help.univention.com/t/32

9.1.3 Configuration and management of Windows desktops

Group policies

Group policies are an Active Directory feature which allows the central configuration of settings for computers and users. Group policies are also supported by Samba/AD domains. The policies only apply to Windows clients; Linux or Mac OS systems cannot evaluate the policies.

Group policies are often referred to as GPOs (*group policy objects*). Put more precisely, a GPO can contain a series of policies. Despite their name, group policy objects cannot be assigned directly to certain user groups, but instead are linked with certain AD administration units (domains, sites or organizational units) in the Samba directory service (Samba AD/DS) and thus refer to subordinate objects. A group-specific or user-specific evaluation is only indirectly possible via the *Security Filtering* of a group policy object, in which the *Apply group policy Allow/Deny* privilege can be directly restricted to certain groups, users or computers.

As a basic rule, a distinction must be made between *group policies* (GPOs) and the similarly named *group policy preferences (GPPs)*:

- The settings made via *GPOs* are binding, whereas *GPPs* are merely used to enter preferences in the registry of Windows clients, which can still be overwritten on the client in certain circumstances.
- The settings made via *GPOs* are also dynamically applied to the target objects, whereas, in contrast, the settings made via *GPPs* are entered statically in the registry of Windows clients (this is also referred to as *tattooing*).

For these reasons, *GPOs* are preferable to *GPPs* in the majority of cases. This remainder of this section deals exclusively with *GPOs*.

In contrast to UCS policies (see *Policies* (page 74)), group policies are not configured via UMC modules, but instead are configured in a separate editor, the *Group Policy Management* editor, which is a component of the *Remote Server Administration Tools (RSAT)*. The installation is described in *Installation of Group Policy Management* (page 175).

There are two types of policies:

User policies
: *User policies* configure a user's settings, e.g., the configuration of the desktop. It is also possible to configure applications via group policies (e.g., the start page of a browser or settings in LibreOffice).

Computer policies
: *Computer policies* define a Windows client's settings.

Computer policies are evaluated for the first time the computer starts up; user policies during login. The policies are also continually evaluated for logged in users / running systems and updated (every 90-120 minutes by default. The period varies at random to avoid peak loads.)

The command `gpupdate /force` can also be run specifically to start the evaluation of group policies.

Some policies - e.g., for the installation of software or for login scripts - are only evaluated during login (user policies) or system startup (computer policies).

The majority of group policies only set one value in the Windows registry, which is then evaluated by Windows or an application. As standard users cannot modify any settings in the corresponding section of the Windows registry, it is also possible to configure restricted user desktops in which, for example, users cannot open the Windows Task Manager.

The group policies are stored in the SYSVOL share, see *Synchronization of the SYSVOL share* (page 173). They are linked with user and host accounts in the Samba directory service.

Installation of Group Policy Management

`Group Policy Management` can be installed as a component of the *Remote Server Administration Tools* on Windows clients. They can be found at Remote Server Administration Tools (RSAT) for Windows 10[41].

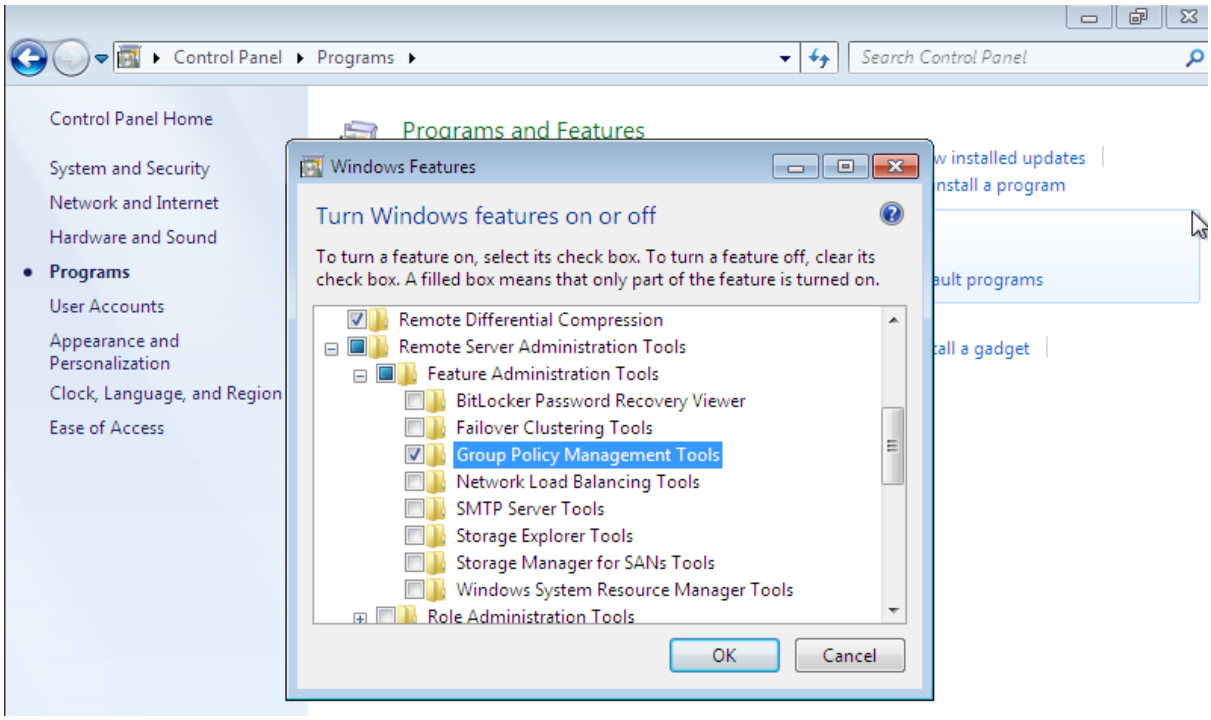


Fig. 9.1: Activating the Group Policy Management tools

Following the installation, Group Policy Management must still be enabled in the Windows Control Panel. This is done by enabling the *Group Policy Management Tools* option under *Start ▸ Control Panel ▸ Programs ▸ Turn Windows features on or off ▸ Remote Server Administration Tools ▸ Feature Administration Tools*.

Following the enabling, Group Policy Management can be run under *Start ▸ Administrative Tools ▸ Group Policy Management*.

Configuration of policies with Group Policy Management

Group policies can only be configured by users who are members of the `Domain Admins` group (e.g., the `Administrator`). When logging in, attention must be paid to logging in with the domain Administrator account and not the local Administrator account. Group Policy Management can be run on any system in the domain.

If more than one Samba domain controller is in use, consideration must be given to the replication of the GPO data, see *Configuration of group policies in environments with more than one Samba domain controller* (page 178).

There are two basic possibilities for creating GPOs:

- They can be created in the *Group Policy Objects* folder and then linked to different positions in the LDAP. This is practical if a policy is to be linked to several positions in the LDAP.
- The GPO can also be created at an LDAP position ad hoc and then directly linked to it. This is the simpler means for small and medium-sized domains. Domains created ad hoc are also shown in the *Group Policy Objects* folder.

A policy can have one of three statuses: `enabled`, `disabled` or `unset`. The effect is always based on the formulation of the policy. For example, if it says *Disable feature xy*, the policy must be enabled to switch off the

[41] https://www.microsoft.com/en-us/download/details.aspx?id=45520

feature. Some policies have additional options, for example the *Enable mail quota* policy could include an additional option for managing the storage space.

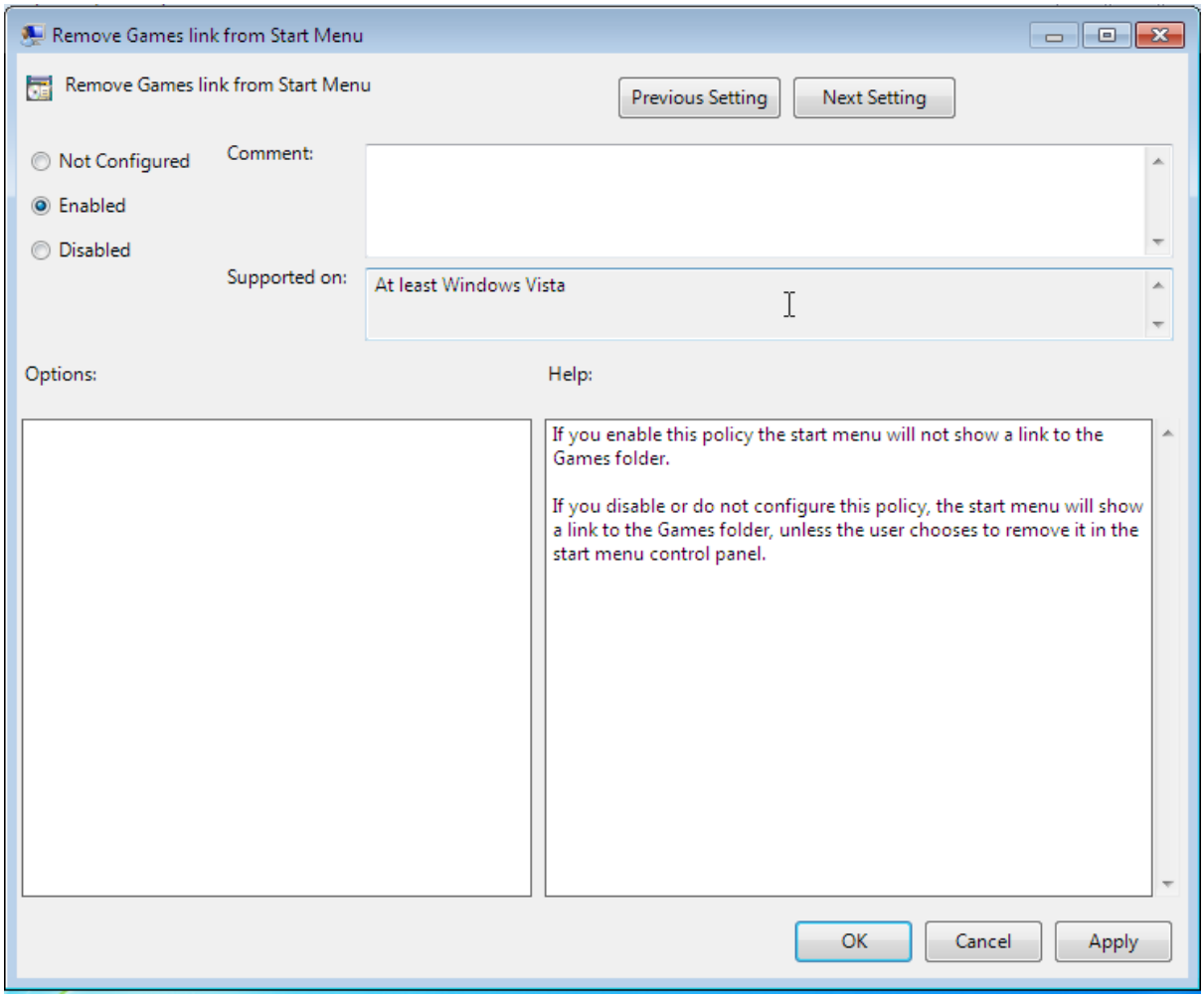


Fig. 9.2: Editing a policy

Two standard policy objects are predefined:

Default Domain Policy
: The *Default Domain Policy* object can be used to configure global policies for all users and computers within the same domain.

Default Domain Controllers Policy
: The *Default Domain Controllers Policy* object has no use in a Samba domain (in a Microsoft AD domain the policies for Microsoft domain controllers would be performed via this object). The configuration of the Samba domain controllers in UCS is largely performed via Univention Configuration Registry.

AD domains can be structured in sites. All the sites are listed in the main menu of *Global Policy Management*. There is also a list of the domains there. The current Samba versions do not support forest domains, so there is only ever one domain displayed here.

One domain can be structured in different organizational units (OUs). This can, for example, be used to store the employees from accounting and the users in the administration department in different LDAP positions.

Group policies can mutually overlap. In this case, the inheritance principle applies, e.g., the superordinate policies overwrite the subordinate ones. The applicable policies for a user can be displayed on the Windows client either with the modeling wizard in *Group Policy Management* or by entering the command **`gpresult /user USERNAME /v`** in the Windows command line.

The policies are evaluated in the following order:

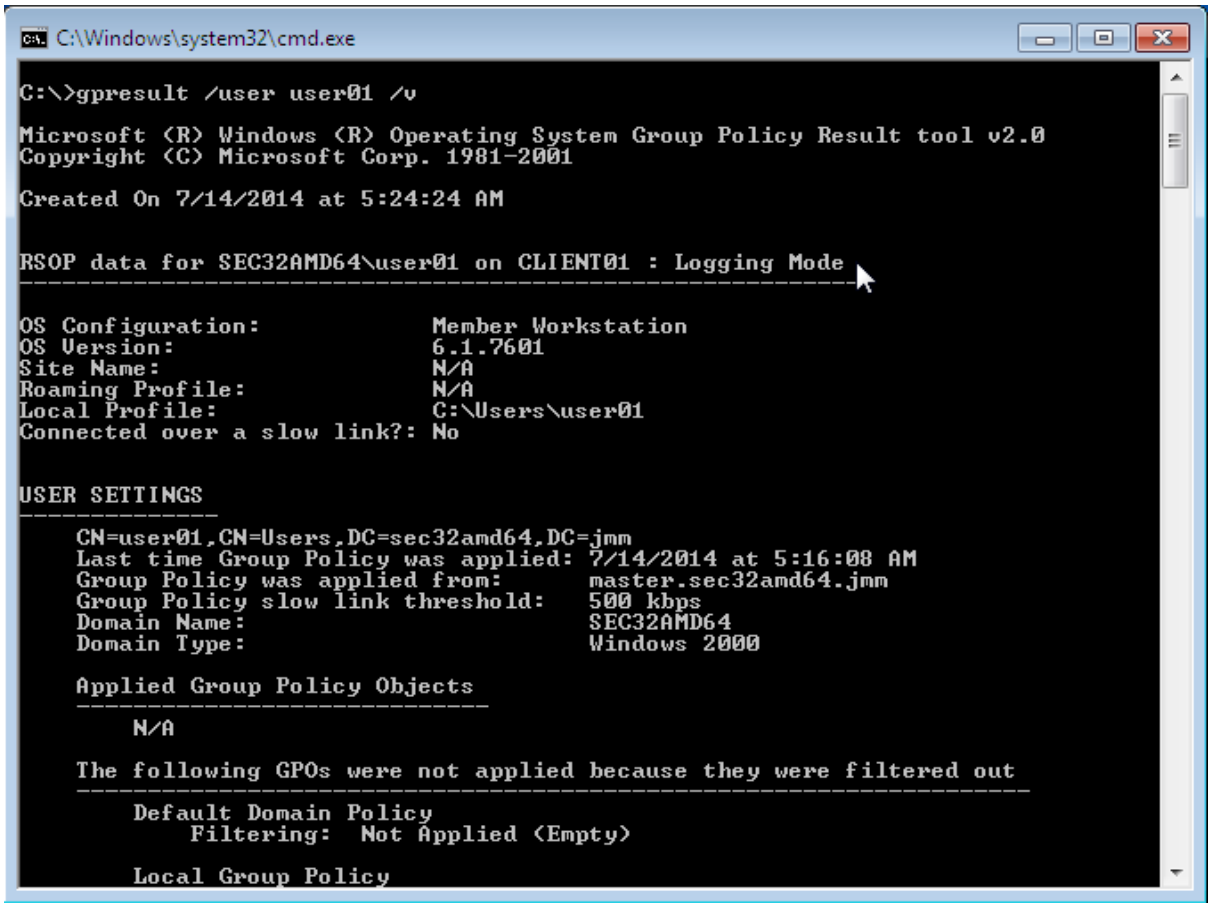


Fig. 9.3: Evaluating the GPO for the user `user01`

1. By default *Default Domain Policy* settings apply for all the users and computers within the domain.
2. Policies linked to an OU overwrite policies from the default domain policy. If the OUs are nested further, in the case of conflict, the "most subordinate" policies in each case, in other words the one most closely linked to the target object, apply. The following evaluation order applies:
   - Assignment of a policy to an Active Directory site
   - Settings of the default domain policy
   - Assignment of a policy to an organizational unit (OU) (in turn, each subordinate OU overrules policies from superordinate OUs).

Example: A company blocks access to the **`Windows Task Manager`** in general. This is done by enabling the *Remove Task Manager* policy in the *Default Domain Policy* object. However, the Task Manager should still be available to some staff with the requisite technical expertise. These users are saved in the *IT staff* OU. An additional group policy object is now created in which the *Remove Task Manager* policy is set to *disabled*. The new GPO is linked with the *IT staff* OU.

Configuration of group policies in environments with more than one Samba domain controller

A group policy is technically composed of two parts: On the one hand there is a directory in the domain controllers' file system which contains the actual policy files which are to be implemented on the Window system (saved in the SYSVOL share (see *Synchronization of the SYSVOL share* (page 173))). On the other hand there is an object with the same name in the LDAP tree of the Samba directory service (Samba AD/DS), which is usually saved below an LDAP container named *Group Policy Objects*.

Although the LDAP replication between the domain controllers is performed in just a few seconds, the files in the SYSVOL share are only replicated every five minutes in the default setting. It must be noted that the application of newly configured group policies in this period may fail if a client happens to consult a domain controller which has not yet replicated the current files.

Administrative templates (ADMX/ADM)

The policies displayed in *Group Policy Management* can be expanded with so-called *administrative templates*. This type of template defines the name under which the policy should appear in Group Policy Management and which value should be set in the Windows registry. Administrative templates are saved in so-called *ADMX files* (previously *ADM files*), see *Group Policy ADMX Syntax Reference Guide* [8].

Among other things, ADMX files offer the advantage that they can be provided centrally across several domain controllers so that Group Policy Management on all Windows clients displays the same configuration possibilities, see *How to Implement the Central Store for Group Policy Admin Templates, Completely (Hint: Remove Those .ADM files!)* [9].

The following example of an ADM file defines a computer policy in which a registry key is configured for the (fictitious) Univention RDP client. ADM files can also be converted to the newer ADMX format using third-party tools. The administrative template must have the file suffix `.adm`:

```
CLASS MACHINE
CATEGORY "Univention"
POLICY "RDP client"
KEYNAME "Univention\RDP\StorageRedirect"
EXPLAIN "If this option it activated, sound output is enabled in the RDP client"
VALUENAME "Sound redirection"
VALUEON "Activated"
VALUEOFF "Deactivated"
END POLICY
END CATEGORY
```

The ADM file can then be converted to the ADMX format or imported directly via Group Policy Management. This is done by following the context menu *Administrative templates* ‣ *Add/Remove Templates* option. *Add* can be used to

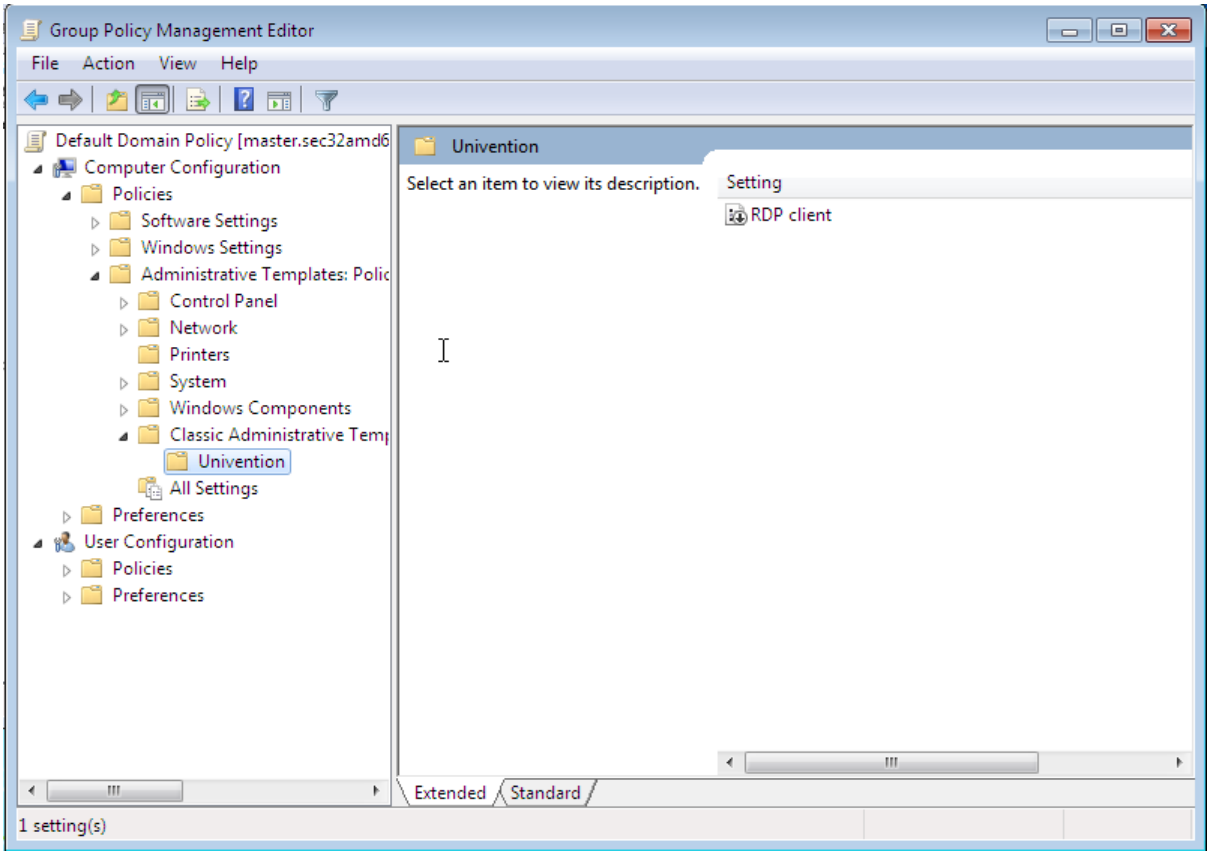


Fig. 9.4: The activated administrative template

import an ADM file. The administrative templates are also saved in the SYSVOL share and replicated, which allows Group Policy Management to access them from the Windows clients.

Application of policies based on computer properties (WMI filters)

It is also possible to configure policies based on system properties. These properties are provided via the Windows Management Instrumentation interface. The mechanism which builds on this is known as *WMI filtering*. This makes it possible, for example, to apply a policy only to PCs with a 64-bit processor architecture or with at least 8 GB of RAM. If a system property changes (e.g., if more memory is installed), the respective filter is automatically re-evaluated by the client.

The WMI filters are displayed in the domain structure in the *WMI Filters* container. *New* can be used to define an additional filter. The filter rules are defined under *Queries*. The rules are defined in a syntax similar to SQL. Examples rules can be found in Microsoft [10] and Heitbrink [11].

Logon scripts / NETLOGON share

The NETLOGON share serves the purpose of providing logon scripts in Windows domains. The logon scripts are executed following after the user login and allow the adaptation of the user's working environment. Scripts have to be saved in a format which can be executed by Windows, such as `bat`.

The logon scripts are stored in `/var/lib/samba/sysvol/`*`Domainname`*`/scripts/` and provided under the share name *NETLOGON*. The filename of the script must be given relative to that directory.

The NETLOGON share is replicated within the scope of the SYSVOL replication.

The logon script can be assigned for each user, see *User management through Univention Management Console module* (page 106).

Configuration of the file server for the home directory

The home directory can be defined user-specifically in the UMC module *Users*, see *User management through Univention Management Console module* (page 106). This is performed with the setting `*Windows home path*, e.g., `\\ucs-file-serversmith`.

The multi edit mode of UMC modules can be used to assign the home directory to multiple users at one time, see *Editing objects* (page 72).

Roaming profiles

Samba supports roaming profiles, i.e., user settings are saved on a central server. This directory is also used for storing the files which the user saves in the *My Documents* folder. Initially, these files are stored locally on the Windows computer and then synchronized onto the Samba server when the user logs off.

No roaming profiles are used by default in Samba/AD.

Roaming profiles can be configured via a group policy found under *Computer configuration* ‣ *Policies* ‣ *Administrative templates* ‣ *System* ‣ *User profiles* ‣ *Set roaming profile path for all users logging onto this computer*. If this is set to the UNC path `%LOGONSERVER%\%USERNAME%\windows-profiles\default` the profile data will get written to the directories `windows-profiles\default.V?` in the home directory of the user located on the currently chosen logon server.

Alternatively the profile path can be defined for individual user accounts. This is possible in the UMC module *Users* under the *Account* tab by filling the field *Windows profile directory*. The corresponding UDM property is called `profilepath`. In the OpenLDAP backend this is stored in the LDAP attribute `sambaProfilePath`.

If the profile path is changed, then a new profile directory will be created. The data in the old profile directory will be kept. These data can be manually copied or moved to the new profile directory. Finally, the old profile directory can be deleted.

Note: As standard, the `Administrator` accesses shares with `root` rights. If as a result the profile directory is created with the root user, it should be manually assigned to the `Administrator` with the command **`chown`**.

9.2 Active Directory Connection

Univention Corporate Server can be operated together with an existing Active Directory domain (AD domain) in two different ways. Both modes can be set up using the **`Active Directory Connection`** application from the Univention App Center (see *Installation of further software* (page 99)). This is available on a Primary Directory Node and Backup Directory Node.

The two modes are:

- UCS as a part (domain member) of an AD domain (see *UCS as a member of an Active Directory domain* (page 181))
- Synchronization of account data between an AD domain and a UCS domain (see *Setup of the UCS AD connector* (page 183)).

In both modes, the **`Active Directory Connection`** service is used in UCS (UCS AD Connector for short), which can synchronize the directory service objects between a Windows 2012/2016/2019/2022 server with Active Directory (AD) and the OpenLDAP directory of Univention Corporate Server.

In the first case, the configuration of a UCS server system as a member of an AD domain, the AD functions as the primary directory service and the respective UCS system joins the trust context of the AD domain. The domain membership gives the UCS system restricted access to the account data of the Active Directory domain. The setup of this operating mode is described in detail in *UCS as a member of an Active Directory domain* (page 181).

The second mode, which can be configured via the **`Active Directory Connection`** app, is used to run the UCS domain parallel to an existing AD domain. In this mode, each domain user is assigned a user account with the

same name in both the UCS and the AD domain. Thanks to the use of the name identity and the synchronization of the encrypted password data, this mode allows transparent access between the two domains. In this mode, the authentication of a user in the UCS domain occurs directly within the UCS domain and as such is not directly dependent on the AD domain. The setup of this operating mode is described in detail in *Setup of the UCS AD connector* (page 183).

9.2.1 UCS as a member of an Active Directory domain

In the configuration of a UCS server system as a member of an AD domain (*AD member* mode), the AD functions as the primary directory service and the respective UCS system joins the trust context of the AD domain. The UCS system is not able to operate as an Active Directory domain controller itself. The domain membership gives the UCS system restricted access to the account data of the Active Directory domain, which it exports from the AD by means of the UCS AD Connector and writes locally in its own OpenLDAP-based directory service. In this configuration, the UCS AD Connector does not write any changes in the AD.

The *AD member* mode is ideal for expanding an AD domain with applications that are available on the UCS platform. Apps installed on the UCS platform can then be used by the users of the AD domain. The authentication is still performed against native Microsoft AD domain controllers.

The setup wizard can be started directly from the UCS installation by selecting *Join into an existing Active Directory domain*. Subsequently, the setup wizard can be installed with the app **`Active Directory Connection`** from the Univention App Center. Alternatively, the software package **`univention-ad-connector`** can be installed. Further information can be found in *Installation of further software* (page 99).

Note:

- The *AD member* mode can only be configured on a Primary Directory Node.
- The name of the DNS domain of the UCS systems must match that of the AD domain. The hostname must of course be different.
- All the AD and UCS servers in a connector environment must use the same time zone.

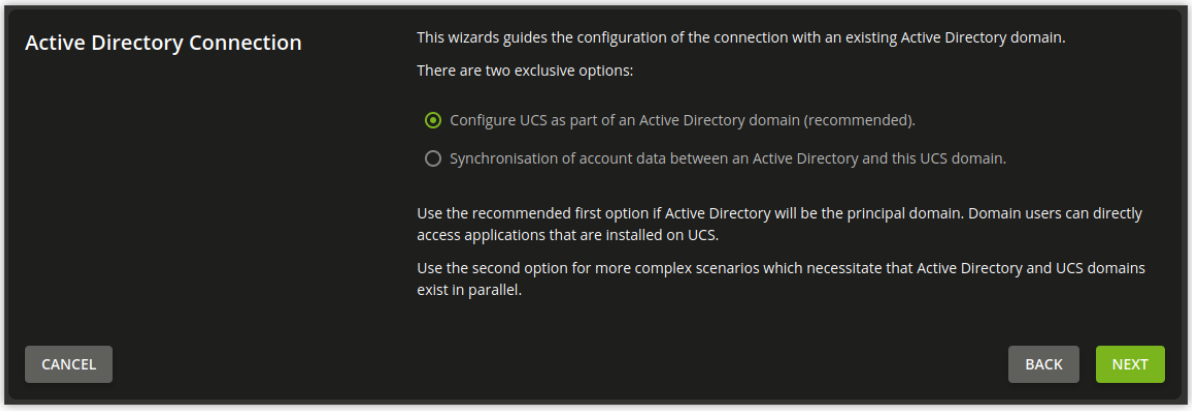


Fig. 9.5: Configuration of the operating mode as part of an AD domain

In the first dialogue window of the setup wizard, the point *Configure UCS as part of an AD domain* is preselected and can be confirmed with *Next*.

The next dialogue window requests the address of an AD domain controller as well as the name of the standard administrator account of the AD domain and its password. The standard AD administrator account should be used here. The specified AD domain controller should also provide DNS services for the domain. Pressing the *Join AD domain* button starts the domain join.

If the system time of the UCS system is more than 5 minutes ahead of the system time of the AD domain controller, manual adjustment of the system times is required. This is necessary because the AD Kerberos infrastructure is used

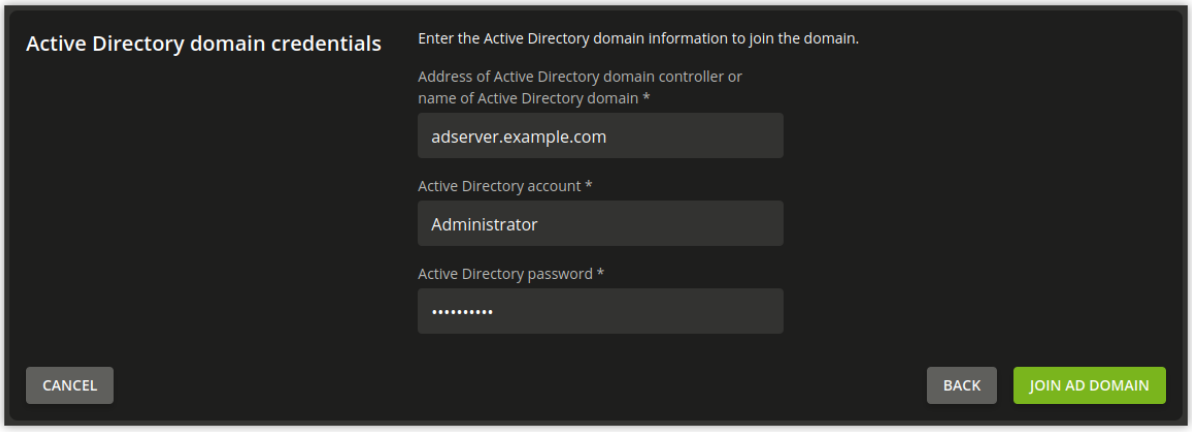


Fig. 9.6: Domain join of an AD domain

for the authentication. System times should not, however, be turned back, in order to avoid inconsistencies.

The domain join is performed automatically. The subsequent dialogue window should be confirmed with *Finish*. Then the UMC server should be restarted by clicking *Restart*.

Note: Once the *AD member* mode has been set up, the authentication is performed against the AD domain controller. **Consequently, the password from the AD domain now applies for the administrator.** If an AD domain with a non-English language convention has been joined, the `administrator` account from UCS is automatically changed to the spelling of the AD during the domain join. The same applies for all user and group objects with *Well Known SID* (e.g., `Domain Admins`).

Warning: If additional UCS systems were already part of the UCS domain in addition to the Primary Directory Node, they must also join the domain anew. At the same time they recognize that the Primary Directory Node is in *AD member* mode and also join the authentication structure of the AD domain and can then also provide Samba file shares, for example.

Note: As the AD Kerberos infrastructure is used for the authentication of users in this mode, it is essential that the system times of UCS and the AD domain controller are synchronized (with a tolerance of 5 minutes). For this purpose, the AD domain controller is configured as the NTP time server in UCS. In the case of authentication problems, the system time should always be the first thing to be checked.

Following this setup, the UMC module *Active Directory Connection* can be used for further administration, e.g., for checking whether the service is running and to restart it if necessary (see *Starting/Stopping the Active Directory Connection* (page 187)).

To use an encrypted connection between Active Directory and the Primary Directory Node not only for the authentication, but also for data exchange itself, the root certificate of the certification authority can be exported from the AD domain controller and uploaded via the UMC module. Further information on this topic is available in *Importing the SSL certificate of the Active Directory* (page 185).

By default the Active Directory connection setup in this way does not transfer any password data from AD to the UCS directory service. Some apps from the Univention App Center require encrypted password data. If an app needs it, a note is shown in the App Center.

In *AD member* mode the UCS AD Connector exports object data from the AD with the authorizations of the Primary Directory Node's machine account by default. These authorizations are not sufficient for exporting encrypted password data. In this case, the LDAP DN of a privileged replication user can be adjusted manually in the Univention Configuration Registry Variable `connector/ad/ldap/binddn` (page 291). This must be a member of

the `Domain Admins` group in the AD. The corresponding password must be saved in a file on the Primary Directory Node and the filename entered in the Univention Configuration Registry Variable *connector/ad/ldap/bindpw* (page 291). If the access password is changed at a later point in time, the new password must be entered in this file. The access rights for the file should be restricted so that only the `root` owner has access.

The following commands demonstrate the steps in an example:

```
$ ucr set connector/ad/ldap/binddn=Administrator
$ ucr set connector/ad/ldap/bindpw=/etc/univention/connector/password
$ touch /etc/univention/connector/password
$ chmod 600 /etc/univention/connector/password
$ echo -n "Administrator password" > /etc/univention/connector/password
$ ucr set connector/ad/mapping/user/password/kinit=false
```

If needed, the AD domain controller can also be replaced by the Primary Directory Node at a later point in time. This is possible via the **Active Directory Takeover** application (see *Migrating an Active Directory domain to UCS using Univention AD Takeover* (page 196)).

9.2.2 Setup of the UCS AD connector

As an alternative to membership in an AD domain, as described in the previous section, the **Active Directory Connection** can be used to synchronize user and group objects between a UCS domain and an AD domain. In addition to unidirectional synchronization, this operating mode also allows bidirectional synchronization. In this operating mode, both domains exist in parallel and their authentication systems function independently. The prerequisite for this is the synchronization of the encrypted password data.

By default containers, organizational units, users, groups and computers are synchronized.

The UCS AD connector can only be installed on a Primary Directory Node or Backup Directory Node system.

Information on the attributes configured in the basic setting and particularities to take into account can be found in *Details on preconfigured synchronization* (page 194).

The identical user settings in both domains allow users to access services in both environments transparently. After logging in to a UCS domain, subsequent connection to a file share or to an Exchange server with Active Directory is possible without a renewed password request. Users and administrators will find users and groups of the same name on the resources of the other domain and can thus work with their familiar permission structures.

The initialization is performed after the first start of the connector. All the entries are read out of the UCS, converted to AD objects according to the mapping set and added (or modified if already present) on the AD side. All the objects are then exported from the AD and converted to UCS objects and added/modified accordingly on the UCS side. As long as there are changes, the directory service servers continue to be requested. The UCS AD connector can also be operated in a unidirectional mode.

Following the initial sync, additional changes are requested at a set interval. This value is set to five seconds and can be adjusted manually using the Univention Configuration Registry variable *connector/ad/poll/sleep* (page 292).

If an object cannot be synchronized, it is firstly reset (*rejected*). Following a configurable number of cycles – the interval can be adjusted using the Univention Configuration Registry variable *connector/ad/retryrejected* (page 292) – another attempt is made to import the changes. The standard value is ten cycles. In addition, when the UCS AD Connector is restarted, an attempt is also made to synchronize the previously rejected changes again.

Basic configuration of the UCS AD Connector

The UCS AD Connector is configured using a wizard in the UMC module *Active Directory Connection*.

The module can be installed from the Univention App Center with the application **Active Directory Connection**. Alternatively, the software package **univention-ad-connector** can be installed. Additional information can be found in *Installation of further software* (page 99).

Note: All AD and UCS servers in a connector environment must use the same time zone.

Warning: Despite intensive tests it is not possible to rule out that the results of the synchronization may affect the operation of a productive domain. The connector should therefore be tested for the respective requirements in a separate environment in advance.

It is convenient to perform the following steps with a web browser from the AD domain controller, as the files need to be downloaded from the AD domain controller and uploaded to the wizard.

In the first dialog window of the setup wizard, the point *Synchronization of content data between an AD and this UCS domain* must be selected and confirmed with *Next*.

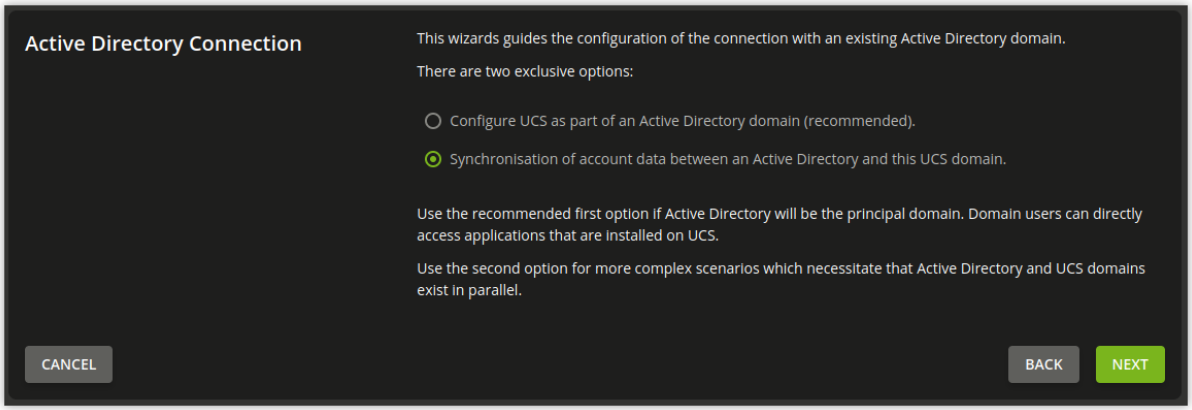


Fig. 9.7: Configuration of the UCS AD Connector via UMC module

The address of an AD domain controller is requested in the next dialogue window. Here you can specify the IP address of a fully qualified DNS name. If the UCS system is not be able to resolve the computer name of the AD system, the AD DNS server can either be configured as the DNS forwarder under UCS or a DNS host record can be created for the AD system in the UMC module *DNS* (see *A/AAAA records (host records)* (page 217)).

Alternatively, a static entry can also be adopted in `/etc/hosts` via Univention Configuration Registry, e.g.

```
$ ucr set hosts/static/192.0.2.100=w2k8-32.ad.example.com
```

In the *Active Directory account* field, the user is configured which is used for the access on the AD. The setting is saved in the Univention Configuration Registry Variable `connector/ad/ldap/binddn` (page 291). The replication user must be a member of the `Domain Admins` group in the AD.

The password used for the access must be entered in the *Active Directory password* field. On the UCS system it is only saved locally in a file which only the `root` user can read.

Changing the AD access password (page 188) describes the steps required if these access data need to be adjusted at a later point in time.

Clicking on *Next* prompts the setup wizard to check the connection to the AD domain controller. If it is not possible to create an SSL/TLS-encrypted connection, a warning is emitted in which you are advised to install a certification authority on the AD domain controller. It is recommended to follow this advice.

UCS 5.0 requires TLS 1.2, which needs to be activated manually for Windows Server Releases prior to 2012R2. UCS 5.0 doesn't support the hash algorithm SHA-1 any longer. If this has been used in the creation of the AD root certificate or for the certificate of the Windows server then they should be replaced.

Following this step, the setup can be continued by clicking *Next* again. If it is still not possible to create an SSL/TLS-encrypted connection, a security query appears asking whether to set up the synchronization without SSL encryption. If this is needed, the setup can be continued by clicking *Continue without encryption*. In this case, the synchronization of the directory data is performed unencrypted.

If the AD domain controller supports SSL/TLS-encrypted connections, the setup wizard offers *Upload AD root certificate* in the next step. This certificate must be exported from the AD certification authority in advance (see *Importing the SSL certificate of the Active Directory* (page 185)). In contrast, if this step is skipped, the certificate can also be uploaded via the UMC module at a later point in time and the SSL/TLS encryption enabled (until that point all directory data will, however, be synchronized unencrypted).

The connector can be operated in different modes, which can be selected in the next dialogue window *Configuration of Active Directory domain synchronization*. In addition to bidirectional synchronization, replication can also be performed in one direction from AD to UCS or from UCS to AD. Once the mode has been selected, *Next* needs to be clicked.

Once *Next* is clicked, the configuration is taken over and the UCS AD Connector started. The subsequent dialogue window needs to be closed by clicking on *Finish*.

Following this setup, the UMC module *Active Directory Connection* can be used for further administration of the Active Directory Connection, e.g., for checking whether the service is running and restart it if necessary (see *Starting/Stopping the Active Directory Connection* (page 187)).

Note: The connector can also synchronize several AD domains within one UCS domain; this is documented in *Extended Windows integration documentation* [7].

Importing the SSL certificate of the Active Directory

A SSL certificate must be created on the Active Directory system and the root certificate exported to allow encrypted communication. The certificate is created by the Active Directory's certificate service. The necessary steps depend on the Windows versions used. Three versions are shown below as examples.

The encrypted communication between the UCS system and Active Directory can also be deactivated by setting the Univention Configuration Registry Variable `connector/ad/ldap/ssl` (page 291) to `no`. This setting does not affect the replication of encrypted password data.

Exporting the certificate on Windows 2012 / 2016 / 2019 / 2022

If the certificate service isn't installed yet, add it to your domain with the following steps before you proceed:

1. Open the *Server Manager*.
2. Select the role *Active Directory Certificate Services* in *Manage ▸ Add Roles and Features*.
3. In the services list, select *Certification Authority*. The top bar in the *Server Manager* shows a yellow warning triangle.
4. Select the option *Configure Active Directory Certificate Services on the server* and configure the *Certification Authority* as selected role service.
5. Choose *Enterprise CA ▸ Root CA* as type of installation.
6. Click *Create a new private key*, confirm the suggested encryption settings and the name of the certification authority.
7. Choose any period for validity and use the standard paths for the database location.
8. Finally, restart your Windows Active Directory server to let the changes come into effect.

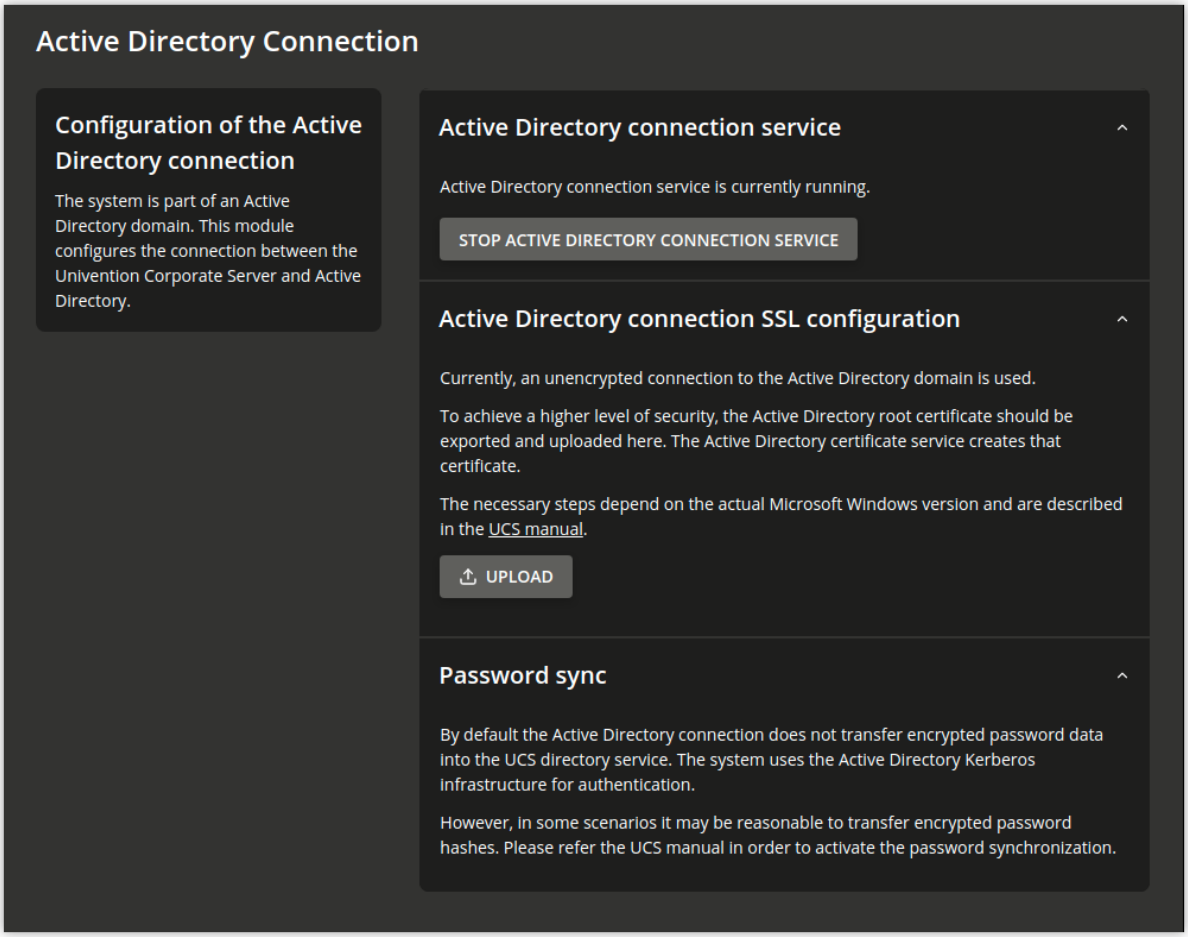


Fig. 9.8: Administration dialogue for the Active Directory Connection

See also:

Install the Certification Authority[Page 187, 42]
for detailed procedure about installing the certificate authority in *Install the Certification Authority* [12].

To export the certificate authority certificate, use the following steps:

1. Open the *Server Manager*.
2. Select the role *Active Directory Certificate Services* (AD CS).
3. Right-click the name of the Windows server and select *Certification Authority*. A window with the certification authority opens. A tree of hosts below *Certification Authority* shows up on the left side.

   Every host has the elements *Revoked Certificates*, *Issued Certificates*, *Pending Requests*, *Failed Requests*, and *Certificate Templates* underneath.
4. In the server list, right-click the Windows host that serves your certificate authority and select *Properties*. Don't mix it up with one of the other elements.
5. In the *Properties* window, select *General* ‣ *CA certificates* ‣ *Certificate #0* and click *View Certificate*.

   Important: It's important to copy the certificate usually with the name `Certificate #0`, because **AD Connection** needs exactly this certificate for a secure connection.
6. In the opening *Certificate* window, select the tab *Details* and click *Copy to File* ….

Copying the Active Directory certificate to the UCS system

The SSL AD certificate should now be imported into the UCS system using the UMC module.

This is done by clicking on *Upload* in the sub menu *Active Directory connection SSL configuration*. This opens a window in which a file can be selected, which is being uploaded and integrated into the UCS AD Connector.

Starting/Stopping the Active Directory Connection

The connector can be started using *Start Active Directory connection service* and stopped using *Stop Active Directory connection service*. Alternatively, the starting/stopping can also be performed with the `/etc/init.d/univention-ad-connector` init-script.

Functional test of basic settings

The correct basic configuration of the connector can be checked by searching in Active Directory from the UCS system. Here one can search e.g. for the administrator account in Active Directory with:

```
$ univention-adsearch cn=Administrator
```

As **univention-adsearch** accesses the configuration saved in Univention Configuration Registry, this allows you to check the reachability/configuration of the Active Directory access.

[42] https://learn.microsoft.com/en-us/windows-server/networking/core-network-guide/cncg/server-certs/install-the-certification-authority

Changing the AD access password

The access data required by the UCS AD Connector for Active Directory are configured via the Univention Configuration Registry Variable *connector/ad/ldap/binddn* (page 291) and *connector/ad/ldap/bindpw* (page 291). If the password has changed or you wish to use another user account, these variables must be adapted manually.

The Univention Configuration Registry Variable *connector/ad/ldap/binddn* (page 291) is used to configure the LDAP DN of a privileged replication user. This must be a member of the `Domain Admins` group in the AD. The corresponding password must be saved locally in a file on the UCS system, the name of which must be entered in the Univention Configuration Registry Variable *connector/ad/ldap/bindpw* (page 291). The access rights for the file should be restricted so that only the `root` owner has access. The following commands show this as an example:

```
$ eval "$(ucr shell)"
$ echo "Updating ${connector_ad_ldap_bindpw?}"
$ echo "for AD sync user ${connector_ad_ldap_binddn?}"
$ touch "${connector_ad_ldap_bindpw?}"
$ chmod 600 "${connector_ad_ldap_bindpw?}"
$ echo -n "Current AD Syncuser password" > "${connector_ad_ldap_bindpw?}"
```

9.2.3 Additional tools / Debugging connector problems

The **Active Directory Connection** provides the following tools and log files for diagnosis:

univention-adsearch
: This tool facilitates a LDAP search in Active Directory. Objects deleted in AD are always shown (they are still kept in an LDAP sub tree in AD). As the first parameter the script awaits an LDAP filter; the second parameter can be a list of LDAP attributes to be displayed.

  Example:

```
$ univention-adsearch cn=administrator cn givenName
```

univention-adconnector-list-rejected
: This tool lists the DNs of non-synchronized objects. In addition, in so far as temporarily stored, the corresponding DN in the respective other LDAP directory will be displayed. In conclusion `lastUSN` shows the ID of the last change synchronized by AD.

  This script may display an error message or an incomplete output if the AD connector is in operation.

remove_ad_rejected.py
: You can use this script to remove an AD object from the AD rejected list located in the internal database file `/etc/univention/connector/internal.sqlite`.

  Example:

```
$ /usr/share/univention-ad-connector/remove_ad_rejected.py \
   -c connector <AD object DN>
```

remove_ucs_rejected.py
: You can use this script to remove an UCS directory object from the UCS rejected list located in the internal database file `/etc/univention/connector/internal.sqlite`.

  Example:

```
$ /usr/share/univention-ad-connector/remove_ucs_rejected.py \
   -c connector <UCS object DN>
```

resync_object_from_ad.py
: You can use this script to re-synchronize directory objects from AD to UCS. Use it to synchronize a single or multiple directory objects.

Example:

```
# to re-sychronize a single object
$ /usr/share/univention-ad-connector/resync_object_from_ad.py \
    -c connector <object DN>

# to re-synchronize all objects matching a specific filter
$ /usr/share/univention-ad-connector/resync_object_from_ad.py \
    -c connector \
    --filter "(objectClass=posixAccount)"

# to re-synchronize all objects matching a specific base
$ /usr/share/univention-ad-connector/resync_object_from_ad.py \
    -c connector \
    --filter "(objectClass=posixAccount)" \
    --base "dc=example,dc=com"
```

resync_object_from_ucs.py

You can use this script to re-synchronize directory objects from UCS to AD. Use it to synchronize a single or multiple directory objects.

Examples:

```
# to re-synchronize a single object
$ /usr/share/univention-ad-connector/resync_object_from_ucs.py \
   -c connector <object DN>

# to re-synchronize all objects matching a specific filter
$ /usr/share/univention-ad-connector/resync_object_from_ucs.py \
   -c connector \
   --filter "<LDAP filter>" \

# to re-synchronize all objects matching a specific base
$ /usr/share/univention-ad-connector/resync_object_from_ucs.py \
   -c connector \
   --filter "<LDAP filter>" \
   --base "<base dn>" \
```

prepare-new-instance

You can use this script to create AD connection instances. The script copies the required files and sets specific UCR variables.

Alternatively, you can use this script to delete an AD connection instance. The script then internally removes the files for the instance and resets the UCR variables.

well-known-sid-object-rename

You can use this script to rename users and groups with well-known SIDs in UDM. The AD Connection uses it to rename users and groups with well-known SIDs.

make-deleted-objects-readable-for-this-machine

You can use this script to grant list and read access to `CN=Deleted Objects` in Active Directory.

Log files

For troubleshooting when experiencing synchronization problems, corresponding messages can be found in the following files on the UCS system:

- `/var/log/univention/connector-ad.log`
- `/var/log/univention/connector-ad-status.log`

9.2.4 Selective synchronization

You can configure the **Active Directory Connection** to synchronize only a specific selection of source objects. You can select the source objects according to the following criteria, described in detail in the following sections:

- Selecting objects by location in the LDAP subtree
- Selecting objects by matching an LDAP filter
- Selecting all items except by location in the LDAP subtree
- Selecting all items except by matching an LDAP filter

Allow only specific LDAP subtrees

To configure the connector to synchronize only specific subtrees of the LDAP structure you can use the following UCR variables:

connector/ad/mapping/allowsubtree/.*/ucs

For synchronization from UCS LDAP directory to Active Directory

Use this Univention Configuration Registry Variable to define a DN from your UCS LDAP directory for the synchronization to the connected Active Directory. Then the *AD Connection* only considers UCS LDAP objects for synchronization that locate in subtrees specified by one of these UCR variables. You must include the LDAP base in the DNs and the comparison of the DNs is case-insensitive.

See the explanation of the `.*` placeholder below.

For example:

```
$ ucr set connector/ad/mapping/allowsubtree/school1/ucs="ou=school1,dc=ucs,
↪domain"
$ ucr set connector/ad/mapping/allowsubtree/school2/ucs="ou=school2,dc=ucs,
↪domain"
```

connector/ad/mapping/allowsubtree/.*/ad

For synchronization from Active Directory to UCS LDAP directory

Use this Univention Configuration Registry Variable to define a DN from your Active Directory for the synchronization to your UCS LDAP directory. Then the *AD Connection* only considers Active Directory objects for synchronization that locate in subtrees specified by one of these UCR variables. You must include the LDAP base in the DNs and the comparison of the DNs is case-insensitive.

See the explanation of the `.*` placeholder below.

For example:

```
$ ucr set connector/ad/mapping/allowsubtree/school1/ad="ou=school1,dc=ad,domain
↪"
$ ucr set connector/ad/mapping/allowsubtree/school2/ad="ou=school2,dc=ad,domain
↪"
```

Placeholder `.*`

The `.*` part of the variable is a placeholder that you can use as an individual label for each variable. If you follow this approach, you create a series of UCR variables of the types described. Each variable contains only one DN.

For each LDAP subtree that you want to allow for synchronization, you have to configure a separate Univention Configuration Registry Variable.

After you have defined or changed the UCR variables, you must restart the **Active Directory Connection**.

Tip: The **Active Directory Connection** determines the position of the target object by dynamic and static factors such as the mapping property attributes `dn_mapping_function` and `position_mapping`, if they're configured in the mapping for individual object types. The position of the corresponding target object can therefore be outside the subtrees corresponding to the Univention Configuration Registry Variable.

Warning: If you use of the `.../allowsubtree/.*/[ad|ucs]` configuration and move a source object from inside a considered subtree to a position that's outside of the combined scope of all of your `.../allowsubtree/.*/[ad|ucs]` definitions, then the **AD Connection** removes the object from the target directory.

Allow only objects that match an LDAP filter

You can configure an LDAP filter for each type of object. **Active Directory Connection** synchronizes only LDAP objects that match that filter. It ignores all other LDAP objects.

For bi-directional synchronization, the filter must match both, the UCS object and the AD object. If an object matching the filter is deleted, the connector also deletes the corresponding object on the other side.

connector/ad/mapping/{type}/allowfilter

The connector only synchronizes those objects with `{type}` object type that match this LDAP filter. `{type}` can be one of the following values:

- `user`
- `group`
- `container`
- `ou`
- `windowscomputer`

For example:

```
$ ucr set connector/ad/mapping/user/allowfilter="(description=sync)"
```

After changing these settings you must restart the **Active Directory Connection**.

Note: However, this filter doesn't support the full LDAP filter syntax. It's always case-sensitive. You can only use the placeholder `*` as a single value without any other characters.

Important: If an object that matches the filter is changed so that the filter no longer matches, the connector **doesn't** synchronize the change. This means that the connector still applies changes from the other side to the object.

If you want to turn off the synchronization for an object, you must make the change on both sides, UCS and Active Directory.

Ignore objects from specific LDAP subtrees

To configure the connector to ignore objects from certain LDAP subtrees you can use the following Univention Configuration Registry Variable:

connector/ad/mapping/ignoresubtree/.*

The variable defines the locations in the directory service that the connector excludes from the synchronization. The values can contain positions in Active Directory and in the UCS LDAP directory. By default, the variable isn't set.

For example:

```
$ ucr set connector/ad/mapping/ignoresubtree/ignore1="cn=alumni,dc=ucs,domain"
$ ucr set connector/ad/mapping/ignoresubtree/ignore2="cn=alumni,dc=ad,domain"
```

After changing this setting you must restart the **Active Directory Connection**.

Ignore objects by LDAP filter

To exclude objects from the synchronization, you can add their names to the following Univention Configuration Registry Variable:

connector/ad/mapping/{type}/ignorelist

The connector **doesn't** synchronize the objects that this variable defines as values. Separate multiple values by commas. For the possible values for `{type}`, see Table 9.1. The table also shows which LDAP attributes you need to consider in the filter depending on the object type.

Table 9.1: Mapping for which `{type}` needs which LDAP attribute

{type}	Value from LDAP attribute
user	uid
group	cn
container	cn
ou	ou
windowscomputer	cn

For example, the `user` type considers the LDAP attribute `uid`:

```
$ ucr set connector/ad/mapping/user/ignorelist="Administrator,krbtgt,root,
↪pcpatch,mmustermann"
```

Important: Some of the `ignorelist` settings have defaults that are important for the functionality of the connector. Make sure that you don't overwrite these settings. You can verify the current value of a Univention Configuration Registry Variable with the following command:

```
$ ucr get connector/ad/mapping/user/ignorelist
```

For more flexibility you can also set an LDAP filter to ignore objects. Use the following Univention Configuration Registry Variable:

connector/ad/mapping/{type}/ignorefilter

The connector **doesn't** synchronize the objects that match this LDAP filter. `{type}` can be have one of the following values:

- `user`
- `group`
- `container`

- `ou`
- `windowscomputer`

For example:

```
$ ucr set connector/ad/mapping/user/ignorefilter="(description=no sync)"
```

Note: However, this filter doesn't support the full LDAP filter syntax. It's always case-sensitive. You can only use the placeholder `*` as a single value without any other characters.

After changing these settings, you must restart the **`Active Directory Connection`**.

Priority of allow and ignore rules

This section describes the processing order for the previously documented settings for selective synchronization.

The **`Active Directory Connection`** processes the allow and ignore rules in a defined order. Depending on the evaluation result, the connector behaves as follows:

- If a rule results in the connector ignoring an object, the connector stops processing the rule and doesn't synchronize the object.
- If a rule results in the connector not ignoring an object, the connector evaluates the next rule. If the rule was the last rule and there's no next rule, the connector synchronizes the object.

The connector evaluates the rules for each object in the following order:

1. **Allow subtree**:

   UCR variables
   connector/ad/mapping/allowsubtree/./ucs* (page 190) and *connector/ad/mapping/allowsubtree/.*/ad* (page 190)

   No match
   No synchronization. Stop.

   Match
   Continue.

2. **Allow filter**:

   UCR variable
   connector/ad/mapping/{type}/allowfilter (page 191)

   No match
   No synchronization. Stop.

   Match
   Continue.

3. **Ignore subtree**:

   UCR variable
   *connector/ad/mapping/ignoresubtree/.** (page 192)

   No match
   Continue.

   Match
   No synchronization. Stop.

4. **Ignore filter**:

UCR variables
connector/ad/mapping/{type}/ignorelist (page 192) and *connector/ad/mapping/{type}/ignorefilter* (page 192)

No match
Continue.

Match
No synchronization. Stop.

5. **End of rules**.
6. **Synchronize object**.

9.2.5 Details on preconfigured synchronization

By default, the **Active Directory Connection** excludes some LDAP subtrees from the synchronization. You can find the list of ignored subtrees in the `/var/log/univention/connector-ad-mapping.log` file under the `ignore_subtree` setting for each object type.

Containers and organizational units

Containers and organizational units are synchronized together with their description. In addition, the `cn=mail` and `cn=kerberos` containers are ignored on both sides. Some particularities must be noted for containers on the AD side. In the *User manager* Active Directory offers no possibility to create containers, but displays them only in the advanced mode (*View ▸ Advanced settings*).

Take the following particularities into account:

- Containers or organizational units deleted in AD are deleted recursively in UCS, which means that any non-synchronized subordinate objects, which are not visible in AD, are also deleted.

Groups

Groups are synchronized using the group name, whereby a user's primary group is taken into account (which is only stored for the user in LDAP in AD).

Group members with no opposite in the other system, e.g., due to ignore filters, are ignored (thus remain members of the group).

The description of the group is also synchronized.

Particularities

Take the following particularities into account:

- The *pre Windows 2000 name* (LDAP attribute `samAccountName`) is used in AD, which means that a group in Active Directory can appear under a different name from in UCS.
- The connector ignores groups, which have been configured as a *Well-Known Group* under *Samba group type* in Univention Directory Manager. There is no synchronization of the SID or the RID.
- Groups which were configured as *Local Group* under *Samba group type* in Univention Directory Manager are synchronized as a *global group* in the Active Directory by the connector.
- Newly created or moved groups are always saved in the same subcontainer on the opposite side. If several groups with the same name are present in different containers during initialization, the members are synchronized, but not the position in LDAP. If one of these groups is migrated on one side, the target container on the other side is identical, so that the DNs of the groups can no longer be differentiated from this point onward.

- Certain group names are converted using a mapping table so that, for example in a German language setup, the UCS group `Domain Users` is synchronized with the AD group *Domänen-Benutzer*. When used in anglophone AD domains, this mapping can result in *germanophone* groups' being created and should thus be deactivated in this case. This can be done using the Univention Configuration Registry Variable *connector/ad/mapping/group/language* (page 291).

  The complete table is:

UCS group	*AD group*
`Domain Users`	`Domänen-Benutzer`
`Domain Admins`	`Domänen-Admins`
`Windows Hosts`	`Domänencomputer`

- Nested groups are represented differently in AD and UCS. In UCS, if groups are members of groups, these objects can not always be synchronized on the AD side and appear in the list of rejected objects. Due to the existing limitations in Active Directory, nested groups should only be assigned there.
- If a global group *A* is accepted as a member of another global group *B* in Univention Directory Manager, this membership does not appear in Active Directory because of the internal AD limitations in **Windows 2000/2003**. If group *A*'s name is then changed, the group membership to group *B* will be lost. Since **Windows 2008** this limitation no longer exists and thus global groups can also be nested in Active Directory.

Custom mappings

For custom mappings, see Active Directory Connection custom mappings[43] in *Univention Developer Reference* [3].

Users

Users are synchronized like groups using the username or using the AD pre Windows 2000 name. The *First name*, *Last name*, *Primary group* (in so far as present on the other side), *Organization*, *Description*, *Street*, *City*, *Postal code*, *Windows home path*, *Windows login script*, *Disabled* and *Account expiry date* attributes are transferred. Indirectly *Password*, *Password expiry date* and *Change password on next login* are also synchronized. *Primary email address* and *Telephone number* are prepared, but commented out due to differing syntax in the mapping configuration.

The `root` and `Administrator` users are exempted.

Take the following particularities into account:

- Users are also identified using the name, so that for users created before the first synchronization on both sides, the same process applies as for groups as regards the position in LDAP.
- In some cases, a user to be created under AD, for which the password has been rejected, is deleted from AD immediately after creation. The reasoning behind this is that AD created this user firstly and then deletes it immediately once the password is rejected. If these operations are transmitted to UCS, they are transmitted back to AD. If the user is re-entered on the AD side before the operation is transmitted back, it is deleted after the transmission. The occurrence of this process is dependent on the polling interval set for the connector.
- AD and UCS create new users in a specific primary group (usually `Domain Users` or `Domänen-Benutzer`) depending on the presetting. During the first synchronization from UCS to AD the users are therefore always a member in this group.

[43] https://docs.software-univention.de/developer-reference/5.0/en/misc.html#ad-connection-custom-mappings

9.3 Migrating an Active Directory domain to UCS using Univention AD Takeover

UCS supports the takeover of user, group and computer objects as well as Group Policy Objects (GPOs) from a Microsoft Active Directory (AD) domain. Windows clients do not need to rejoin the domain. The takeover is an interactive process consisting of three distinct phases:

1. Copy all objects from Active Directory to UCS
2. Copy of the group policy files from the AD server to UCS
3. Deactivate the AD server and assign all FSMO roles to the UCS DC

The following requirements must be met for the takeover:

- The UCS Directory Node (Primary Directory Node) needs to be installed with a unique hostname, not used in the AD domain.
- The UCS Directory Node needs to be installed with the same DNS domain name, NetBIOS (pre Windows 2000) domain name and Kerberos realm as the AD domain. It is also recommended to configure the same LDAP base DN.
- The UCS Directory Node needs to be installed with a unique IPv4 address in the same IP subnet as the Active Directory domain controller that is used for the takeover.

Caution: If the system is already a member of an Active Directory Domain, installing the *Active Directory Takeover* application removes this membership. Therefore, the installation of the *Takeover* application has to take place only shortly before the actual takeover of the AD domain.

The *Active Directory Takeover* application must be installed from the Univention App Center for the migration. It must be installed on the system where the Univention S4 Connector is running (see *Univention S4 connector* (page 173), usually the Primary Directory Node).

9.3.1 Preparation

The following steps are strongly recommended before attempting the takeover:

- A backup of the AD server(s) should be performed.
- If user logins to the AD server are possible (e.g. through domain logins or terminal server sessions) it is recommended to deactivate them and to stop any services in the AD domain, which deliver data, e.g. mail servers. This ensures that no data is lost in case of a rollback to the original snapshot/backup.
- It is recommended to set the same password for the `Administrator` account on the AD server as the corresponding account in the UCS domain. In case different passwords are used, the password that was set last, will be the one that is finally valid after the takeover process (timestamps are compared for this).
- In a default installation the `Administrator` account of the AD server is deactivated. It should be activated in the local user management module.

The activation of the `Administrator` account on the AD server is recommended because this account has all the required privileges to copy the GPO SYSVOL files. The activation can be achieved by means of the *Active Directory Users and Computers* module or by running the following two commands:

```
> net user administrator /active:yes
> net user administrator PASSWORD
```

9.3.2 Domain migration

The takeover must be initiated on the UCS Directory Node that runs the Univention S4 Connector (by default the Primary Directory Node). During the takeover process Samba must only run on this UCS system. If other UCS Samba/AD Nodes are present in the UCS domain, Samba needs to be stopped on those systems. This is important to avoid data corruption by mixing directory data taken over from Active Directory with Samba/AD directory data replicated from other UCS Samba/AD Nodes.

Other UCS Samba/AD systems can be stopped by logging into each of the other UCS Directory Nodes as the `root` user and running

```
$ /etc/init.d/samba4 stop
```

After ensuring that only the Univention S4 Connector host runs Samba/AD, the takeover process can be started. If the UCS domain was installed initially with a UCS version before UCS 3.2, the following Univention Configuration Registry Variable needs to be set first:

```
$ ucr set connector/s4/mapping/group/grouptype=false
```

The takeover is performed with the UMC module *Active Directory Takeover*. The IP address of the AD system must be specified under *Name or address of the Domain Controller*. An account from the AD domain must be specified under *Active Directory Administrator account* which is a member of the AD group `Domain Admins` (e.g., the `Administrator`) and the corresponding password entered under *Active Directory Administrator password*.

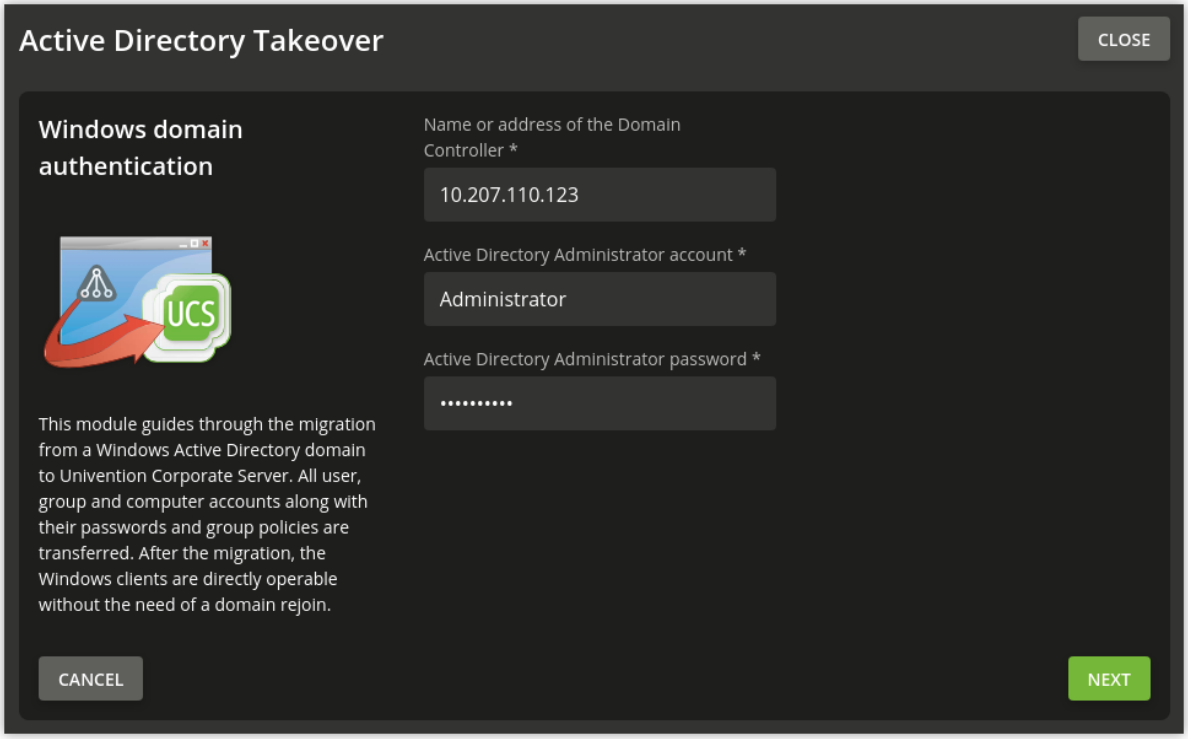


Fig. 9.9: First phase of domain migration

The module checks whether the AD domain controller can be accessed and displays the domain data to be migrated.

When *Next* is clicked, the following steps are performed automatically:

1. Adjust the system time of the UCS system to the system time of the Active Directory domain controller in case the UCS time is behind by more than three minutes.
2. Join the UCS Directory Node into the Active Directory domain.
3. Start Samba and the Univention S4 connector to replicate the Active Directory objects into the UCS OpenLDAP directory.



Fig. 9.10: Overview of the data to be migrated

4. When "*Well Known*" account and group objects (identified by their special RIDs) are synchronized into the UCS OpenLDAP, a listener module running on each UCS system sets a Univention Configuration Registry variable to locally to map the English name to the non-English AD name.

   These variables are used to translate the English names used in the UCS configuration files to the specific names used in Active Directory. To give an example, if `Domain Admins` has a different name in the AD, then the Univention Configuration Registry variable *groups/default/domainadmins* (page 294) is set to that specific name (likewise for uses, e.g. *users/default/administrator* (page 306)).

Additional information is logged to `/var/log/univention/ad-takeover.log` as well as to `/var/log/univention/management-console-module-adtakeover.log`.

The UCS Directory Node now contains all users, groups and computers of the Active Directory domain. In the next step, the SYSVOL share is copied, in which among other things the group policies are stored.

This phase requires the login to the Active Directory domain controller as the `Administrator` (or the equivalent non-English name). There a command needs to be started to copy the group policy files from the Active Directory SYSVOL share to the UCS SYSVOL share.

The command to be run in shown in the UMC module. If it has been successfully run, it must be confirmed with *Next*.

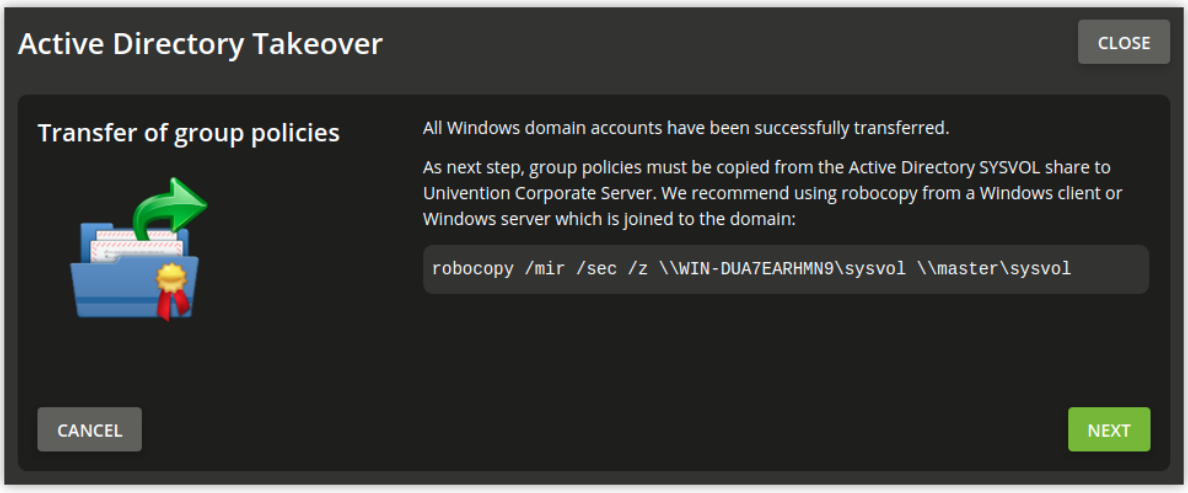


Fig. 9.11: Copying the SYSVOL share

It may be necessary to install the required **robocopy** tool, which is part of the Windows Server 2003 Resource Kit Tools. Starting with Windows 2008 the tool is already installed.

Note: The `/mir` option of **robocopy** mirrors the specified source directory to the destination directory. Please be aware that if you delete data in the source directory and execute this command a second time, this data will also be deleted in the destination directory.

After successful completion of this step, it is now necessary to shutdown all domain controllers of the Active Directory domain. Then *Next* must be clicked in the UMC module.

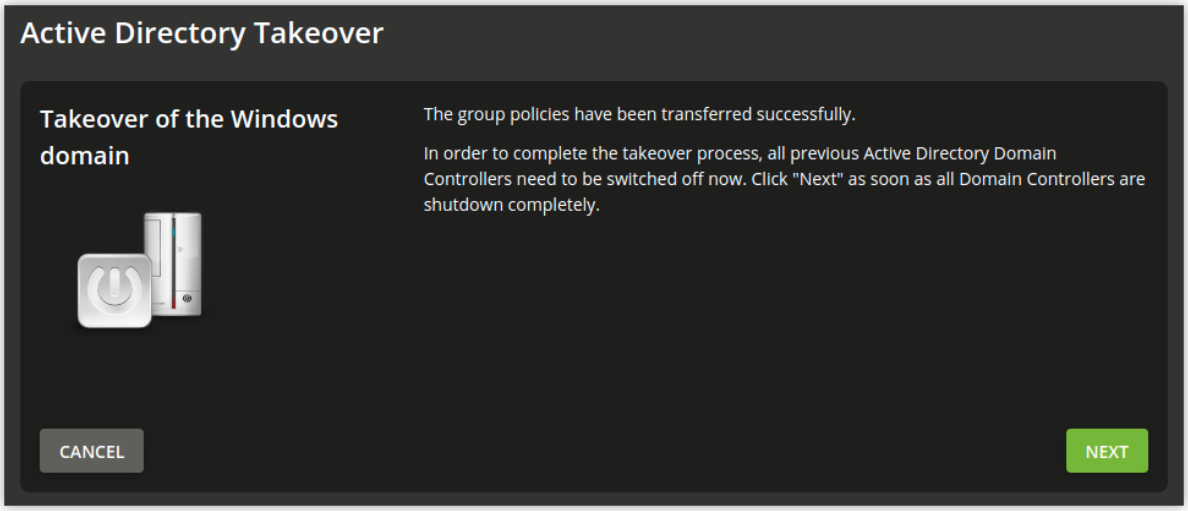


Fig. 9.12: Shutdown of the AD server(s)

The following steps are now automatically performed:

1. Claiming all FSMO roles for the UCS Directory Node. These describe different tasks that a server can take on in an AD domain.
2. Register the name of the Active Directory domain controller as a DNS alias (see *CNAME record (Alias records)* (page 217)) for the UCS DNS server.
3. Configure the IP address of the Active Directory domain controller as a virtual Ethernet interface.
4. Perform some cleanup, e.g. removal of the AD domain controller account and related objects in the Samba SAM account database.
5. Finally restart Samba and the DNS server.

9.3.3 Final steps of the takeover

Finally the following steps are required:

1. The domain function level of the migrated Active Directory domain needs to be checked by running the following command:

```
> samba-tool domain level show
```

   In case this command returns the message `ATTENTION: You run SAMBA 4 on a forest function level lower than Windows 2000 (Native)`. the following commands should be run to fix this:

```
> samba-tool domain level raise --forest-level=2003 --domain-level=2003
> samba-tool dbcheck --fix --yes
```

2. In case there has been more than one Active Directory domain controller in the original Active Directory domain, all the host accounts of the other domain controllers must be removed in the computers management UMC modules. In addition their accounts must be removed from the Samba SAM database. This may be done by logging in to a migrated Windows client as member of the group `Domain Admins` and running the tool **`Active Directory Users and Computers`**.
3. If more than one UCS Directory Node with Samba/AD has been installed, these servers need to be re-joined.
4. All Windows clients need to be rebooted.

9.3.4 Tests

It is recommended to perform thorough tests with Windows client systems, e.g.

- Login to a migrated client as a migrated user.
- Login to a migrated client as the *Administrator*.
- Test group policies.
- Join of a new Windows client.
- Create a new UCS user and login to a Windows client.

9.4 Trust relationships

Trust relationships between domains make it possible for users from one domain to sign in to computers from another domain.

In general, Windows trust relations can be *unidirectional* or *bidirectional*. Technically a bidirectional trust is simply realized as two unidirectional trusts, one in each direction.

The terminology of unidirectional trusts depends on the perspective of either the trusting or trusted domain: From the perspective of the trusting domain, the trust is called *outgoing*. From the perspective of the trusted domain, the trust is called *incoming*.

In UCS, outgoing trust (UCS trusts Windows) is not supported. As a consequence, bidirectional trust is not supported either.

When setting up and using the trust relationship the domain controllers of both domains must be able to reach each other over the network and identify each other via DNS. At least the fully qualified DNS names of the domain controllers of the respective remote domain must be resolvable to allow communication between both domains to work. This can be achieved by configuring conditional DNS forwarding in both domains.

The following example assumes, that the UCS Samba/AD DC Primary Directory Node `primary.ucsdom.example` has the IP address `192.0.2.10` and that the Active Directory domain controller `dc1.addom.example` of the remote domain has the IP address `192.0.2.20`.

On the UCS side the conditional forwarding of DNS queries can be set up as `root` with the following commands:

```
$ cat >>/etc/bind/local.conf.samba4 <<__EOT__
zone "addom.example" {
  type forward;
  forwarders { 192.0.2.20; };
};
__EOT__
$ systemctl restart bind9
```

The success can be checked by running:

```
$ host dc1.addom.example
```

In addition, it may be useful to create a static entry for the domain controller of the remote Active Directory domain in the file `/etc/hosts`:

```
$ ucr set hosts/static/192.0.2.20=dc1.addom.example
```

On a Windows AD DC, a so-called *conditional forwarding* can be set up for the UCS domain via the DNS server console.

Trust relationships can only be configured on domain controllers but they affect the whole domain.

After this preliminary work, the trust relationship can be established directly from the command line of the UCS Samba/AD DC using the tool **samba-tool**:

```
$ samba-tool domain trust create addom.example \
  -k no -UADDOM\\Administrator%ADAdminPassword \
  --type=external --direction=incoming
```

The trust can be checked using the following commands:

```
$ samba-tool domain trust list
$ wbinfo --ping-dc -domain=addom.example
$ wbinfo --check-secret -domain=addom.example
```

After the setup, a UCS user should be able to sign in to systems of the remote Active Directory domain. Users must either use the format `UCSDOM\username` as login name or their Kerberos principal in the notation `username@ucsdom.example`.

CHAPTER

TEN

IDENTITY MANAGEMENT CONNECTION TO CLOUD SERVICES

UCS offers an integrated Identity Management System. Through Univention Management Console, users and groups among others can be administered. Depending on the installed services, these identities are made available through different interfaces e.g. via LDAP.

The management system can be extended with the help of provided extensions, also called apps. Thus users or groups can also be replicated in cloud services. In the App Center there are also among others extensions for Microsoft 365 or G Suite.

Thanks to single sign-on (SSO), users can sign in with their usual password and immediately get to work online in the cloud. The password remains in the company's network and is not transferred to the cloud service.

The following chapter describes how to set up the Microsoft 365 Connector and the Google Apps for Work Connector.

10.1 Microsoft 365 Connector

The synchronization of users and groups and teams to an Azure Directory Domain, which will then be used by Microsoft 365, is made possible by the **`Microsoft Office Connector`**. The connector makes it possible to control which of the users created in UCS can use Microsoft 365. The selected users will be provisioned accordingly into the Azure Active Directory domain. It is configurable which user attributes are synchronized and which are anonymized during synchronization.

The single sign-on login to Microsoft 365 is done via the UCS integrated SAML implementation. Authentication takes place against the UCS server, and no password hashes are transmitted to Microsoft Azure Cloud. The user's authentication is done exclusively via the client's web browser. The web browser should however be able to resolve the DNS records of the UCS domain, this is a particularly important point to note for mobile devices.

10.1.1 Setup

To setup the Microsoft 365 Connector a Microsoft 365 Administrator account, a corresponding Account in the Azure Active Directory, as well as a Domain verified by Microsoft[44] are required. The first two are provided for test purposes by Microsoft for free. However to configure the SSO, a separate internet domain where TXT records can be created is required.

In case there is no Microsoft 365 subscription available, one can be configured it via https://www.office.com/ in the *trial for business* section. A connection is not possible with a private Microsoft account.

You should then sign in with a *Microsoft 365 Administrator Account* into the *Microsoft 365 Admin Center*. At the bottom left of the navigation bar select *Azure AD* to open the *Azure Management Portal* in a new window.

In the *Azure Active Directory* section the menu item *Custom domain names* can be used to add and verify your own domain. For this it is necessary to create a TXT record in the DNS of your own domain. This process can take up to several minutes. Afterwards the *status* of the configured domain will be displayed as **Verified**.

[44] https://learn.microsoft.com/en-us/entra/fundamentals/add-custom-domain

Now the **Microsoft 365 Connector** App can be installed from the App Center on the UCS system. The installation takes a few minutes. There is a setup wizard available for the initial configuration. After completing the wizard the connector is ready for use.

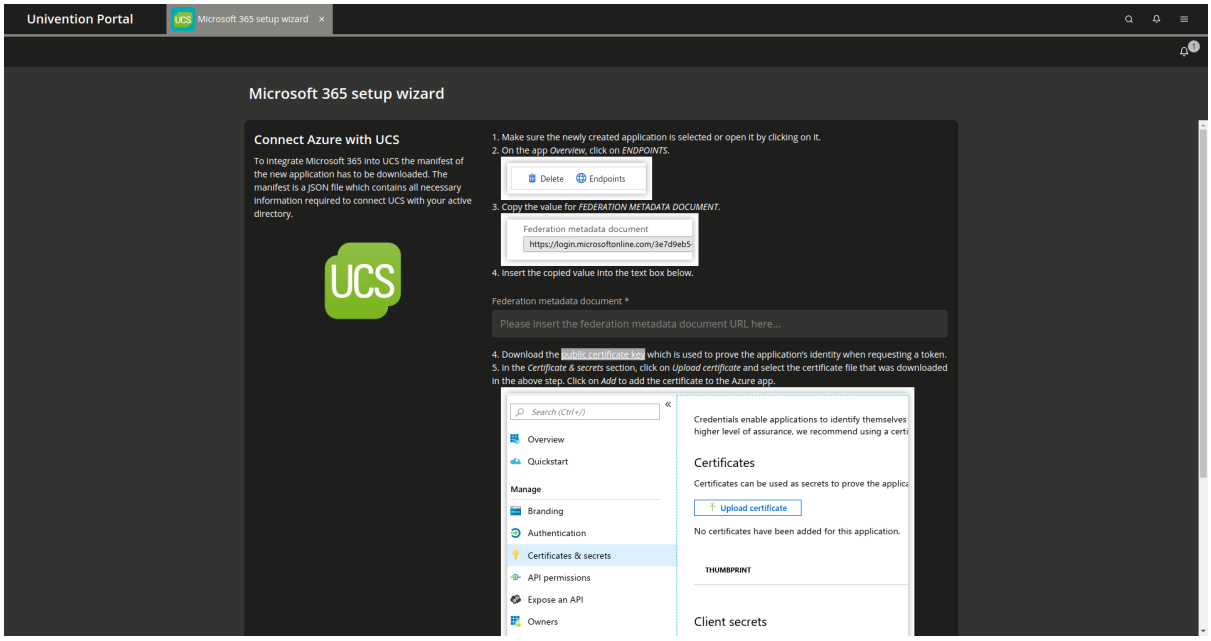


Fig. 10.1: Microsoft 365 Setup assistant

10.1.2 Configuration

After the end of the installation through the setup wizard, users can be enabled to use Microsoft 365. This configuration can be done through the user module on each user object on the *Microsoft 365* tab. Usage and allocation of licenses are acknowledged in the *Microsoft 365 Admin Center*.

Users

If a change is made to the user, the changes are likewise replicated to the Azure Active Directory domain. There is no synchronization from the Azure Active Directory to the UCS system. This means changes made in Azure Active Directory or Office Portal may be overridden by changes to the same attributes in UCS.

Due to Azure Active Directory security policies, users or groups in the Azure AD can't be deleted during synchronization. They are merely disabled and renamed. The licenses are revoked in the Azure Active Directory so that they become available to other users. Users and groups whose names start with `ZZZ_deleted` can be deleted in *Microsoft 365 Admin Center*.

It is necessary to configure a country for the user in Microsoft 365. The connector uses the specification of the country from the contact data of the user. If not set, it uses the setting of the server. With the help of Univention Configuration Registry Variable *office365/attributes/usageLocation* (page 301) a 2-character abbreviation, e.g. `US`, can be set as the default.

Through Univention Configuration Registry Variable *office365/attributes/sync* (page 301), the LDAP attributes (e.g. first name, last name, etc.) of a user's account which will to be synchronized are configured. The form is a comma-separated list of LDAP attributes. Thus adaptation to personal needs is possible.

With the Univention Configuration Registry Variable *office365/attributes/anonymize* (page 301), a comma-separated list of LDAP attributes can be configured that are created in the Azure Active Directory but filled with random values. The Univention Configuration Registry Variables *office365/attributes/static/.** (page 301) allows the filling of attributes on the Microsoft side with a predefined value.

The Univention Configuration Registry Variable *office365/attributes/never* (page 301) can be used to specify a comma separated list of LDAP attributes that should not be synchronized even when they appear in *office365/attributes/sync* (page 301) or *office365/attributes/anonymize* (page 301).

The Univention Configuration Registry Variables *office365/attributes/mapping/.** (page 301) define a mapping of UCS LDAP attributes to Azure Attributes. Usually these variables don't need to be changed. The synchronization of the groups of Microsoft 365 user can be enabled with the Univention Configuration Registry Variable *office365/groups/sync* (page 301).

Changes to Univention Configuration Registry Variables are implemented only after restarting the Univention Directory Listener.

Groups

The **Microsoft 365 Connector** can synchronize groups to Microsoft Azure Active Directory. The connector synchronizes a group, if it contains at least one user that's enabled for *Microsoft 365*. By default, the connector creates the group as a `Security Group` in *Microsoft 365*.

You can enable the synchronization by setting the Univention Configuration Registry Variable *office365/groups/sync* (page 301) to `yes`.

New in version 5.0-7-erratum-1060: With UCS 5.0 erratum 1060[45], you can change the group type to `Microsoft 365 Group` in the UMC on the *Microsoft 365* tab.

If the UCS system with the connector installed doesn't have at least this version level, you can't change the group type.

When you change the group type, the connector deletes the existing group in *Microsoft 365*, and it creates a group with the group type you defined. *Microsoft 365* doesn't allow to change the group type property of an existing group. The connector adds the group members to the group, and if the group is a *Microsoft 365 Team*, the connector also creates the team.

Caution: Changing the group type can effect the group's permissions and settings in *Microsoft 365*. Change group types with caution.

You can change the default group type of a Microsoft 365 group. The following group types are available:

- `Security`
- `Microsoft 365 Group`

To change the group type, you need to modify the default value of the extended attribute `UniventionMicrosoft365GroupType` to one of the available group types:

```
$ udm settings/extended_attribute modify \
    --dn "cn=UniventionMicrosoft365GroupType,cn=custom attributes,cn=univention,
↪$(ucr get ldap/base)" \
    --set default="Microsoft 365 Group"
```

[45] https://errata.software-univention.de/#/?erratum=5.0x1060

Teams

To use Teams, synchronization of groups must be enabled in the Univention Configuration Registry Variable *office365/groups/sync* (page 301) with the value `yes`, and then the Univention Directory Listener service must be restarted. If UCS groups are to be created as teams in Microsoft 365, the groups must be configured as teams on the *Microsoft 365* tab via the *Microsoft 365 Team* checkbox. Furthermore, it is necessary to define an owner of the team on the same tab. Further settings on the team can be made by the team owners directly in the Teams interface. After activating a group as a team, the group members are added to the new team. Provisioning a new team in Microsoft 365 can take a few minutes.

Ensure that users of a team in Azure are provided with a license that includes the use of Teams.

10.1.3 Synchronization of Users in multiple Azure Active Directories

The Microsoft 365 Connector is able to synchronize users to multiple Azure Active Directories. For each user account, multiple Azure AD instances can be assigned, where an account should be created. A user gets a distinct username (*Userprincipalname* or *UPN*) for every assigned Azure AD.

An alias is assigned to each additional Azure AD connection by the administrator. To manage these aliases the program **`/usr/share/univention-office365/scripts/manage_adconnections`** can be used. A new alias is created by calling `/usr/share/univention-office365/scripts/manage_adconnections create <Aliasname>`. This will set the Univention Configuration Registry Variable *office365/adconnection/wizard* (page 301) to the newly created alias. The value of this Univention Configuration Registry Variable defines which connection is configured by the next run of the Microsoft 365 Configuration Wizard.

After creating the alias, the new connection must be configured through the Microsoft 365 Configuration Wizard, as well.

To use single sign-on with multiple Azure AD connections, a new logical SAML Identity Provider is needed for each connection. This is described in *Extended Configuration* (page 48).

The Identity Provider should get the same name as the alias. If another name was chosen, the PowerShell script to configure single sign-on needs to be adjusted manually. For example the Univention Configuration Registry Variable `saml/idp/entityID/supplement/Aliasname=true` needs to be set on all domain controllers responsible for single sign-on.

A UCS user can only use one Microsoft 365 account in one browser session at a time. To change the connection, a logout from Microsoft 365 is necessary.

A default alias for Microsoft 365 enabled users and groups can be set in the Univention Configuration Registry Variable *office365/defaultalias* (page 301). To synchronize them into a different Azure Active Directory the connection alias must be selected explicitly at the user or group object.

10.1.4 Troubleshooting/Debugging

Messages during the setup are logged in `/var/log/univention/management-console-module-office365.log`.

In case of synchronization problems, the log file of the Univention Directory Listener should be examined: `/var/log/univention/listener.log`.

Some actions of the Connector use long-running Azure Cloud operations, especially when using Teams. These operations are logged in the log file `/var/log/univention/listener_modules/ms-office-async.log` The Univention Configuration Registry Variable *office365/debug/werror* (page 301) activates additional debug output.

10.2 Google Apps for Work Connector

Google Apps for Work Connector allows users and groups to synchronize to a G Suite domain. You can control which of the users created in UCS are allowed to use G Suite. The users selected in this way are provisioned accordingly by UCS into the G Suite domain. It can be configured which attributes are synchronized and attributes can be anonymized.

The single sign-on login to G Suite is done via the UCS integrated SAML implementation. Authentication takes place against the UCS server, and no password hashes are transferred to the G Suite domain. The user's authentication is done exclusively via the client's web browser. However, the browser should be able to resolve the DNS records of the UCS domain, which is particularly important for mobile devices.

10.2.1 Setup

To setup the Google Apps for Work Connectors a G Suite Administrator account, a corresponding account in the G Suite domain, and a domain verified[46] by Google are required. The first two will be provided free of charge by Google for testing purposes. However, configuring the SSO requires a separate internet domain where TXT records can be created.

If no G Suite subscription is available yet, it can be configured via Set up Google Workspace for your organization[47]. A connection with a private Gmail account is not possible.

Afterwards, you should sign in with a *G Suite administrator account* in the Admin Console[48]. The domain should now be verified. For this it is necessary to create a TXT record in the DNS of your own domain. This process can take a few minutes.

Now the Google Apps for Work Connector from the App Center can be installed on the UCS system. The installation only takes a few minutes. There is a setup wizard available for the initial configuration. After completing the wizard the connector is ready for use.

10.2.2 Configuration

After the setup via the setup wizard, you can use the user module on each user object on the *Google Apps tab* to configure that this user is provisioned to G Suite.

If a change is made to the user, the changes will also be replicated to the G Suite domain. There is no synchronization from the G Suite domain to the UCS system. This means that changes made in the G Suite domain may be overwritten by changes to the same attributes in UCS.

If the Google Apps property is removed from a user, the user will be deleted from the G Suite domain accordingly.

The Univention Configuration Registry Variables `google-apps/attributes/mapping/.*` (page 294) is used to configure which LDAP attributes (e.g. first name, last name, etc.) of a user account are synchronized. The Univention Configuration Registry Variable and its values reflect the nested data structure of the G Suite user accounts. The names that follow the percentage sign in the values are the attributes in the UCS LDAP. If all Univention Configuration Registry Variable `google-apps/attributes/mapping/.*` (page 294) are removed, no data other than the primary email address is synchronized.

The Univention Configuration Registry Variable `google-apps/attributes/anonymize` (page 294) can be used to specify comma-separated LDAP attributes that are created in the G Suite domain but filled with random values.

The Univention Configuration Registry Variable `google-apps/attributes/never` (page 294) can be used to specify comma-separated LDAP attributes that should not be synchronized, even if they are configured via `google-apps/attributes/mapping/.*` (page 294) or `google-apps/attributes/anonymize` (page 294).

[46] https://support.google.com/a/topic/9196?hl=en
[47] https://support.google.com/a/answer/6365252?hl=en
[48] https://admin.google.com/

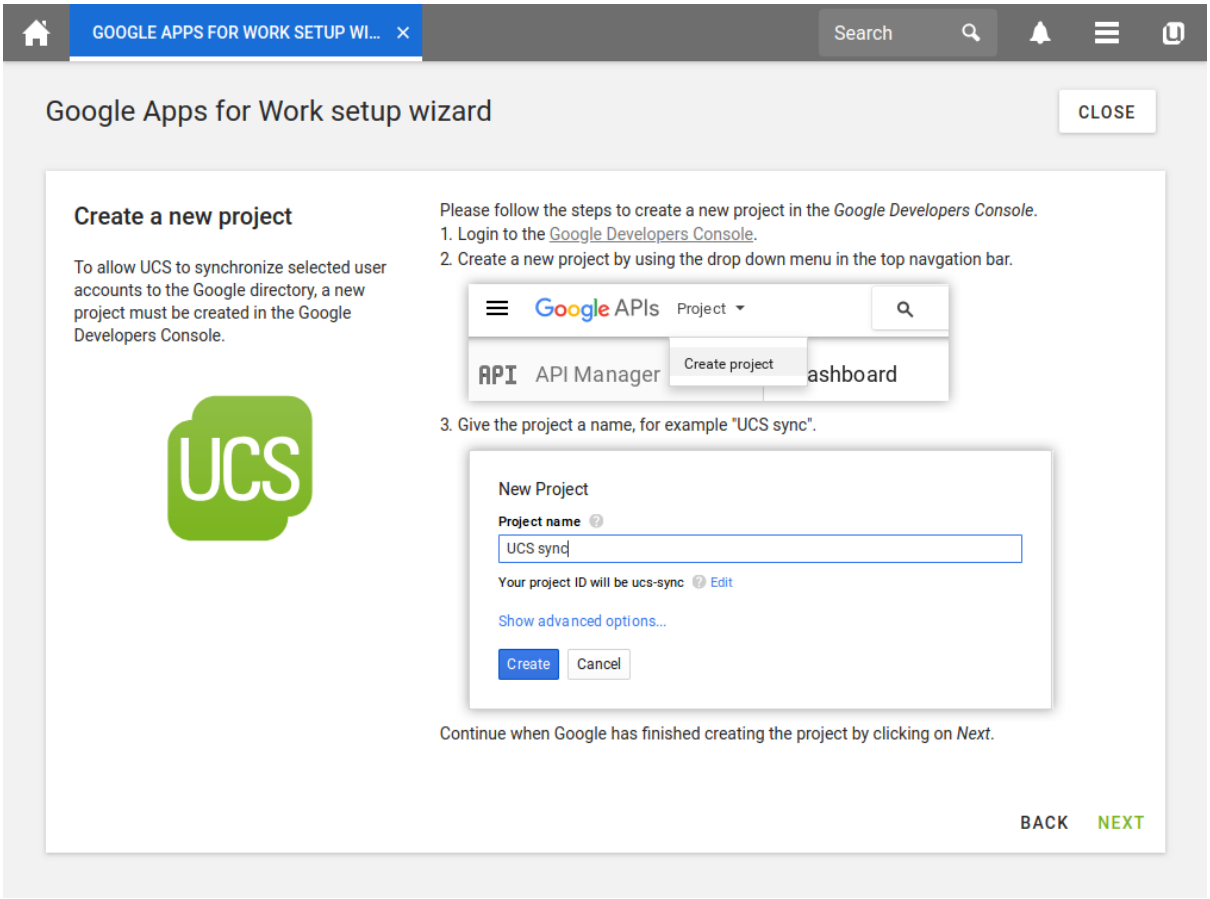


Fig. 10.2: Google Apps for Work Setup Wizard

The synchronization of Google Apps for Work user groups can be enabled with the Univention Configuration Registry Variable *google-apps/groups/sync* (page 294).

Changes to Univention Configuration Registry Variables are implemented after restarting the Univention Directory Listener.

10.2.3 Troubleshooting/Debugging

Messages during setup are logged in the following log file `/var/log/univention/management-console-module-googleapps.log`.

In case of synchronization problems, the log file of the Univention Directory Listener should be checked: `/var/log/univention/listener.log`. The Univention Configuration Registry Variable *google-apps/debug/werror* (page 294) activates additional debug output.

CHAPTER

ELEVEN

IP AND NETWORK MANAGEMENT

This chapter describes how IP addresses for computer systems in a UCS domain can be centrally managed via UMC modules and assigned via DHCP.

Network objects (page 211) bundle available IP address segments of a network. The DNS resolution as well as the assignment of IP addresses via DHCP are integrated in UCS, as detailed in *Administration of DNS data with BIND* (page 212) and *IP assignment via DHCP* (page 220).

Incoming and outgoing network traffic can be restricted via the *Univention Firewall* based on **`iptables`**, see *Packet filter with Univention Firewall* (page 226).

The integration of the proxy server Squid allows the caching of web contents and the enforcement of content policies for web access, see *Web proxy for caching and policy management / virus scan* (page 227).

11.1 Network objects

Network objects can be used to compile available IP addresses; the next available address is then automatically specified during assignment to a computer.

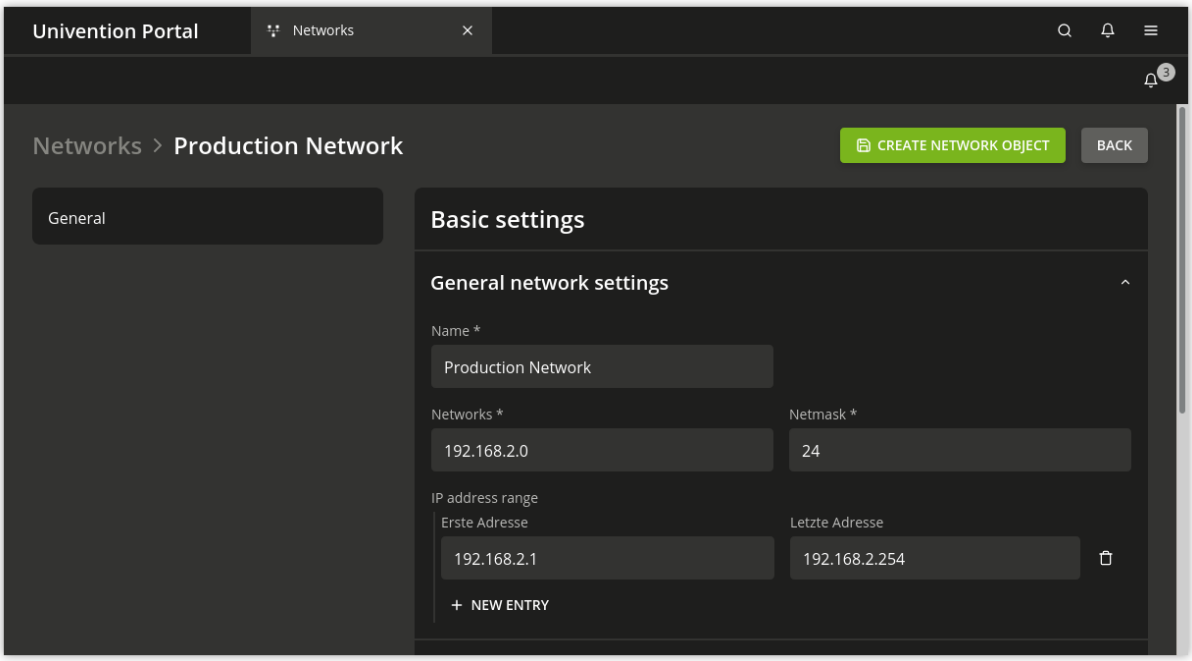


Fig. 11.1: Creating a network object

For example, it is possible to define a network object *Workstation network* which encompasses the IP addresses from `192.0.2.0` to `192.0.2.254`. If a Windows computer object is now created and only the network object selected, an internal check is performed for which IP addresses are already assigned and the next free one selected. This saves

the administrator having to compile the available addresses manually. If a computer object is removed, the address is automatically reassigned.

Network objects are managed in the UMC module *Networks*. For more information about UMC, see *Univention Management Console modules* (page 69).

Table 11.1: *General* tab

Attribute	Description
Name	The name of the network is entered in this input field. This is the name under which the network also appears in the computer management.
Networks	The network address is entered in dot-decimal form in this input field, e.g., `192.0.2.0`.
Netmask	The network mask can be entered in this input field in network prefix or dot-decimal form. If the network mask is entered in dot-decimal form it will be subsequently be converted into the corresponding network prefix and later also shown so.
IP address range	One or more IP ranges can be configured here. When a host is assigned to this network at a later point, it will automatically be assigned the next, free IP address from the IP range entered here. When no IP range is entered here, the system automatically uses the range given by the network and the subnet mark entered. Forward lookup zones and reverse lookup zones can be selected in the sub menu *DNS preferences*. When a host is assigned to this network at a later point, a host record in the forward lookup zone and/or a pointer record in the reverse lookup zone will be created automatically. The zones are also administrated in the UMC module *DNS*, see *Forward lookup zone* (page 214). If no zone is selected here, no DNS records are created during assignment to a computer object. However, the DNS entries can still be set manually.
DNS forward lookup zone	The forward lookup zone where hosts from the network should be added must be specified here. The resolution of the computer name to an IP address is performed via the zone.
DNS reverse lookup zone	The reverse lookup zone where hosts from the network should be added must be specified here. The reverse resolution of the IP address back to a computer name is performed via the zone. A DHCP service can be assigned to the network in the sub menu *DHCP preferences*. When a host is assigned to this network at a later point, a DHCP computer entry with a fixed IP address will be created automatically in the selected DHCP service. The DHCP service settings are also administrated in the UMC module *DHCP*, see *Composition of the DHCP configuration via DHCP LDAP objects* (page 220). If no DHCP service is selected, no DHCP host record is created during assignment to a computer object. However, such an entry can also still be assigned manually.

11.2 Administration of DNS data with BIND

UCS integrates BIND for the name resolution via the domain name system (DNS). The majority of DNS functions are used for DNS resolution in the local domain; however, the UCS BIND integration can also be used for a public name server in principle.

BIND is always available on all UCS Directory Node roles; installation on other system roles is not supported.

The configuration of the name servers to be used by a UCS system is documented in *Network configuration* (page 151).

The following DNS data are differentiated:

Forward lookup zone
A *forward lookup zone* contains information which is used to resolve DNS names into IP addresses. Each DNS zone has at least one authoritative, primary name server whose information governs the zone. Subordinate servers synchronize themselves with the authoritative server via zone transfers. The entry which defines such a zone is called a *SOA record* in DNS terminology.

MX record
The *MX record* of a forward lookup zone represents important DNS information necessary for email routing. It points to the computer which accepts emails for a domain.

TXT records
TXT records include human-readable text and can include descriptive information about a forward lookup zone.

CNAME record
A *CNAME record*, also called an alias record, refers to an existing, canonical DNS name. For example, the actual hostname of the mail server can be given an alias entry *mailserver*, which is then entered in the mail clients. Any number of CNAME records can be mapped to one canonical name.

A record
An *A record* (under IPv6 *AAAA record*) assigns an IP address to a DNS name. *A records* are also known as *Host records* in UCS.

SRV record
A *SRV record*, called a service record in UCS, can be used to save information about available system services in the DNS. In UCS, service records are used among other things to make LDAP servers or the Primary Directory Node known domain-wide.

Reverse lookup zone
A *reverse lookup zone* contains information which is used to resolve IP addresses into DNS names. Each DNS zone has at least one authoritative, primary name server whose information governs the zone, subordinate servers synchronize themselves with the authoritative server via zone transfers. The entry which defines such a zone is the *SOA record*.

PTR record
A *PTR record* (pointer record) allows resolution of an IP address into a hostname. It thus represents the equivalent in a reverse lookup zone of a host record in a forward lookup zone.

11.2.1 Configuration of the BIND name server

Configuration of BIND debug output

The level of detail of the BIND debug output can be configured via the `dns/debug/level` (page 293) and `dns/dlz/debug/level` (page 293) (for the Samba backend, see *Configuration of the data backend* (page 213)) Univention Configuration Registry variables. The possible values are between `0` (no debug tasks) to `11`. A complete list of levels can be found at Liu and Albitz [13].

Configuration of the data backend

In a typical BIND installation on a non-UCS system, the configuration is performed by editing zone files. In UCS, BIND is completely configured via UMC modules, which saves its data in the LDAP directory.

BIND can use two different backend for its configuration:

LDAP backend
The *LDAP backend* accesses the data in the LDAP directory. This is the standard backend. The DNS service is split into two in this case: The *BIND proxy* is the primary name server and uses the DNS standard port `53`. A second server in the background works on port `7777`. If data from the internal DNS zones are edited in the LDAP, the zone file on the second server is updated based on the LDAP information and transmitted to the BIND proxy by means of a zone transfer.

Samba backend
: Samba/AD provides an Active Directory domain. Active Directory is closely connected with DNS, for DNS updates of Windows clients or the localization of NETLOGON shares among other things. If Samba/AD is used, the UCS Directory Node in question is switched over to the use of the *Samba backend*. The DNS database is maintained in Samba's internal LDB database, which Samba updates directly. BIND then accesses the Samba DNS data via the DLZ interface.

When using the Samba backend, a search is performed in the LDAP for every DNS request. With the OpenLDAP backend, a search is only performed in the directory service if the DNS data has changed. The use of the LDAP backend can thus result in a reduction of the system load on Samba/AD systems.

The backend is configured via the Univention Configuration Registry Variable `dns/backend` (page 293). The DNS administration is not changed by the backend used and is performed via UMC modules in both cases.

Configuration of zone transfers

By default the UCS name server allows zone transfers of the DNS data. If the UCS server can be reached from the internet, a list of all computer names and IP addresses can be requested. The zone transfer can be deactivated when using the OpenLDAP backend by setting the Univention Configuration Registry Variable `dns/allow/transfer` (page 293) to `none`.

11.2.2 Administration of DNS data via Univention Management Console module

DNS files are stored in the `cn=dns,base DN` container as standard. Forward and reverse lookup zones are stored directly in the container. Additional DNS objects such as pointer records can be stored in the respective zones.

The relative or fully qualified domain name (FQDN) should always be used in the input fields for computers and not the computer's IP address. A FQDN should always end in a full stop to avoid the domain name being added anew.

The left column of the UMC module *DNS* includes a list of all the forward and reverse lookup zones. To add an object to a zone - for example an alias record to a forward zone - the corresponding zone must be selected. *Add* is then used to create the object in this zone. To create a new forward or reverse zone, start by selecting *All DNS zones*. Clicking on *Add* then creates a new zone. If an object is created within the zone, the zone is labeled in the UMC dialogues as a *superordinate object*.

Forward lookup zone

Forward lookup zones contain information which is used to resolve DNS names into IP addresses. They are managed in the UMC module *DNS* (see *Univention Management Console modules* (page 69)). To add another forward lookup zone, select *All DNS zones* and *Add* ▸ *DNS: Forward lookup zone*.

DNS UMC module forward lookup - General tab

Table 11.2: *General* tab

Attribute	Description
Zone name	This is the complete name of the DNS domain for which the zone will be responsible. The domain name **must not** end in a full stop in zone names!
Zone time to live	The time to live specifies how long these files may be cached by other DNS servers. The value is specified in seconds.
Name servers	The fully qualified domain name with a full stop at the end of the relative domain name of the responsible name server. The first entry in the line is the primary name server for the zone.

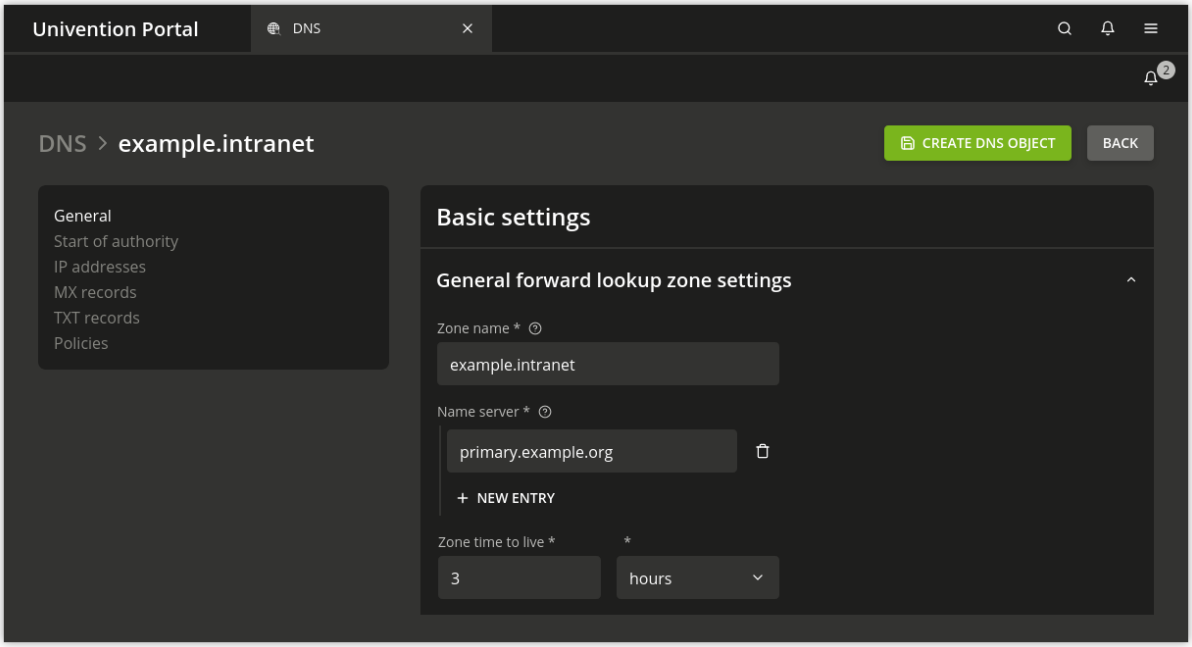


Fig. 11.2: Configuring a forward lookup zone in the UMC module *DNS*

DNS UMC module forward lookup - Start of authority tab

Table 11.3: *Start of authority* tab

Attribute	Description
Contact person	The email address of the person responsible for administrating the zone.
Serial number	Other DNS servers use the serial number to recognize whether zone data have changed. The secondary name server compares the serial number of its copy with that on the primary name server. If the serial number of the secondary is lower than that on the primary, the secondary copies the changed data. There are two commonly used patterns for this serial number: • Start with `1` and increment the serial number with each change. • By including the date the number can be entered in the format `YYYYMMDDNN`, where – `Y` stands for year, – `M` for month, – `D` for day and – `N` for the number of the change of this day. If the serial number is not changed manually, it will be increased automatically with every change.
Refresh interval	The time span in seconds after which the secondary name server checks that its copy of the zone data is up-to-date.
Retry interval	The time span in seconds after which the secondary name server tries again to check that its copy of the zone data is up-to-date after a failed attempt to update. This time span is usually set to be less than the update interval, but can also be equal.
Expiry interval	The time span in seconds after which the copy of the zone data on the secondary becomes invalid if it could not be checked to be up-to-date. For example, an expiry interval of one week means that the copy of the zone data becomes invalid when all requests to update in one week fail. In this case, it is assumed that the files are too outdated after the expiry interval date to be used further. The secondary name server can then no longer answer name resolution requests for this zone.
Negative time to live	The negative time to live specifies in seconds how long other servers can cache no-such-domain (NXDOMAIN) answers. This value cannot be set to more than 3 hours, the default value is 3 hours.

DNS UMC module forward lookup - IP addresses tab

Table 11.4: *IP addresses* tab

Attribute	Description
IP addresses	This input field can be used to specify one or more IP addresses, which are output when the name of the zone is resolved. These IP addresses are queried by Microsoft Windows clients in AD compatible domains.

DNS UMC module forward lookup - MX records tab

Table 11.5: *MX records* tab

Attribute	Description
Priority	A numerical value between 0 and 65535. If several mail servers are available for the MX record, an attempt will be made to engage the server with the lowest priority value first.
Mail server	The mail server responsible for this domain as fully qualified domain name with a full stop at the end. Only canonical names and no alias names can be used here.

DNS UMC module forward lookup - TXT records tab

Table 11.6: *TXT records* tab

Attribute	Description
TXT record	Descriptive text for this zone. Text records must not contain umlauts or other special characters.

CNAME record (Alias records)

CNAME records / alias records are managed in the UMC module *DNS* (see *Univention Management Console modules* (page 69)). To create another record, the forward lookup zone must be selected in the left column. *Add ▸ DNS: Alias record* can be used to create a new record.

Table 11.7: *General* tab

Attribute	Description
Alias	The alias name as fully qualified domain name with a full stop at the end or as a relative domain name which should point to the canonical name.
Canonical name	The canonical name of the computer that the alias should point to, entered as a fully qualified domain name with a full stop at the end or a relative domain name.

A/AAAA records (host records)

Host records are managed in the UMC module *DNS* (see *Univention Management Console modules* (page 69)). To create another record, the forward lookup zone must be selected in the left column. *Add ▸ DNS: Host record* can be used to create a new record.

When adding or editing a computer object a host record can be created automatically or edited.

Table 11.8: *General* tab

Attribute	Description
Hostname	The FQDN with a full stop at the end or the relative domain name of the name server.
IP addresses	The IPv4 and/or IPv6 addresses to which the host record should refer.
Zone time to live	The time to live specifies in seconds how long these files may be cached by other DNS servers.

Service records

Service records are managed in the UMC module *DNS* (see *Univention Management Console modules* (page 69)). To create another record, the forward lookup zone must be selected in the left column. *Add ▸ DNS: Service record* can be used to create a new record.

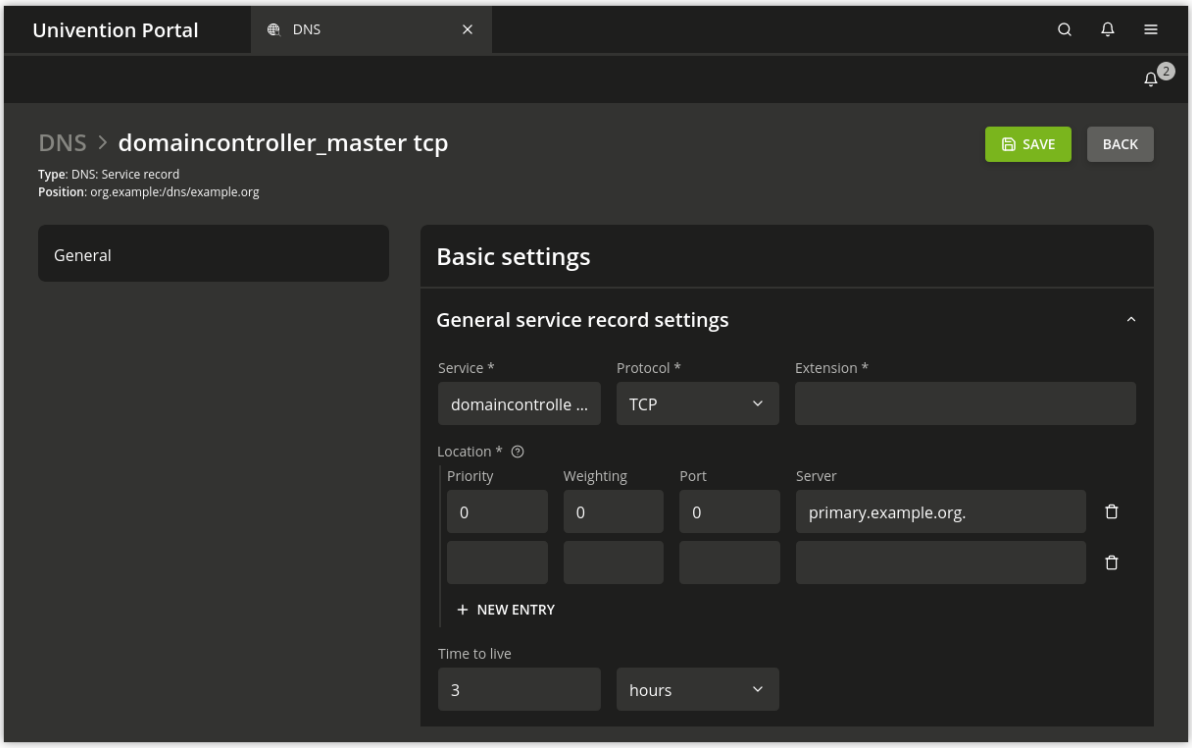


Fig. 11.3: Configuring a service record

A service record must always be assigned to a forward lookup zone and can therefore only be added to a forward lookup zone or a subordinate container.

Table 11.9: *General* tab

Attribute	Description
Service	The name under which the service should be reachable.
Protocol	The protocol via which the record can be accessed (`TCP`, `UDP`, `MSDCS` or `SITES`).
Extension	This input field can be used to specify additional parameters.
Priority	A whole number between 0 and 65535. If more than one server offer the same service, the client will approach the server with the lowest priority value first.
Weighting	A whole number between 0 and 65535. The weight function is used for load balancing between servers with the same priority. When more than one server offer the same service and have the same priority the load is distributed across the servers in relation to the weight function. Example: `Server1` has a priority of `1` and a weight function of `1`, whilst `Server2` also has a priority of `1`, but has a weight function of `3`. In this case, `Server2` will be used three times as often as `Server1`. The load is measured depending on the service, for example, as the number of requests or connection.
Port	The port where the service can be reached on the server (valid value from 1 to 65535).
Server	The name of the server on which the service will be made available, as a fully qualified domain name with a full stop at the end or a relative domain name. Several servers can be entered for each service.
Zone time to live	The time to live specifies how long these files may be cached by other DNS servers.

Reverse lookup zone

A reverse lookup zone is used to resolve IP address into host names. They are managed in the UMC module *DNS*. To add another reverse lookup zone, select *All DNS zones* and *Add ▸ DNS: Reverse lookup zone*.

DNS UMC module reverse lookup - General tab

Table 11.10: *General* tab

Attribute	Description
Subnet	The IP address of the network for which the reverse lookup zone shall apply. For example, if the network in question consisted of the IP addresses `192.0.2.0` to `192.0.2.255`, `192.0.2` should be entered.
Zone time to live	The time to live specifies how long these files may be cached by other DNS servers.

Each DNS zone has at least one authoritative, primary name server whose information governs the zone. Subordinate servers synchronize themselves with the authoritative server via zone transfers. The entry which defines such a zone is called a SOA record in DNS terminology.

DNS UMC module reverse lookup - Start of authority tab

Table 11.11: *Start of authority* tab

Attribute	Description
Contact person	The email address of the person responsible for administrating the zone (with a full stop at the end).
Name servers	The fully qualified domain name with a full stop at the end or the relative domain name of the primary name server.
Serial number	See the documentation on forward lookup zones in *Forward lookup zone* (page 214).
Refresh interval	See the documentation on forward lookup zones in *Forward lookup zone* (page 214).
Retry interval	See the documentation on forward lookup zones in *Forward lookup zone* (page 214).
Expiry interval	See the documentation on forward lookup zones in *Forward lookup zone* (page 214).
Minimum time to live	See the documentation on forward lookup zones in *Forward lookup zone* (page 214).

Pointer record

Pointer records are managed in the UMC module *DNS* (see *Univention Management Console modules* (page 69)). To create another record, the reverse lookup zone must be selected in the left column. *Add ▸ DNS: Pointer record* can be used to create a new record.

Table 11.12: *General* tab

Attribute	Description
Address	The last octet of the computer's IP address (depends on network prefix, see example below).
Pointer	The computer's fully qualified domain name with a full stop at the end. In a network with a 24-bit network prefix (subnet mask `255.255.255.0`) a pointer should be created for the `client001` computer with the IP address `192.0.2.101`. `101` must then be entered in the *Address field* and `client001.company.com.` in *Pointer*. Example: For a network with a 16-bit network prefix (subnet mask `255.255.0.0`) the last two octets should be entered in reverse order for this computer (here `101.1`). `client001.company.com.` also needs to be entered in the *Pointer* field here.

11.3 IP assignment via DHCP

The Dynamic Host Configuration Protocol (DHCP) assigns computers an IP address, the subnet mask and further settings for the gateway or NetBIOS server as necessary. The IP address can be set fixed or dynamic.

The use of DHCP allows central assignment and control of IP addresses via the LDAP directory without performing manual configuration on the individual computer systems.

The DHCP integration in UCS only supports IPv4.

In a *DHCP service*, DHCP servers are grouped in a shared LDAP configuration. Global configuration parameters are entered in the DHCP service; specific parameters in the subordinate objects.

A DHCP server can be installed from the Univention App Center with the application **`DHCP server`**. Alternatively, the software package **`univention-dhcp`** can be installed. Additional information can be found in *Installation of further software* (page 99).

Every DHCP assigns IP addresses via DHCP. In the default setting, only static IP addresses are assigned to computer objects registered in the UCS LDAP.

If only fixed IP addresses are assigned, as many DHCP servers as required may be used in a DHCP service. All the DHCP servers procure identical data from the LDAP and offer the DHCP clients the data multiple times. DHCP clients then accept the first answer and ignore the rest.

If dynamic IP addresses are also assigned, the DHCP failover mechanism must be employed and a maximum of two DHCP servers can be used per subnet.

A *DHCP host* entry is used to make the DHCP service aware of a computer. A DHCP host object is required for computers attempting to retrieve a fixed IP address over DHCP. DHCP computer objects do not normally need to be created manually, because they are created automatically when a DHCP service is assigned to a computer object with a fixed IP address.

A *DHCP subnet* entry is required for every subnet, irrespective of whether dynamic IP addresses are to be assigned from this subnet.

Configuration parameters can be assigned to the different IP ranges by creating *DHCP pools* within subnets. In this way unknown computers can be allowed in one IP range and excluded from another IP range. DHCP pools can only be created below DHCP subnet objects.

If several IP subnets are used in a physical Ethernet network, this should be entered as a *DHCP shared subnet* below a *DHCP shared network*. *DHCP shared subnet* objects can only be created below *DHCP shared network* objects.

Values which are set on a DHCP configuration level always apply for this level and all subordinate levels, unless other values are specified there. Similar to policies, the value which is closest to the object always applies.

11.3.1 Composition of the DHCP configuration via DHCP LDAP objects

The left column of the UMC module *DHCP* includes a list of all the DHCP services. To add an object to a DHCP service - for example in an additional subnet - the corresponding service must be selected. *Add* is then used to create the object in this service. To create a new DHCP service, start by selecting *All DHCP services*. Clicking on *Add* then creates a new service. If an object is saved within a service, the service is labeled in UMC dialogues as a *superordinate object*.

Administration of DHCP services

DHCP services are managed in the UMC module *DHCP* (see *Univention Management Console modules* (page 69)). To create a new DHCP service, *All DHCP services* needs to be selected in the left column of the UMC module. Clicking on *Add* then creates a new service.

A DHCP server can only serve one DHCP service; to use another DHCP service, a separate DHCP server must be set up (see *Administration of DHCP server entries* (page 221)).

The following parameters are often set on the DHCP service object which then apply to all the computers which are served by this DHCP service (unless other values are entered in lower levels):

- *Domain name* and *Domain name servers* under *Policy: DHCP DNS*
- *NetBIOS name servers* under *Policy: DHCP NetBIOS*

A description of this and the other DHCP policies can be found at *Configuration of clients via DHCP policies* (page 224).

Table 11.13: *General* tab

Attribute	Description
Service name	An unambiguous name for the DHCP service must be entered in this input field, e.g., `company.example`.

Administration of DHCP server entries

Each server which should offer the DHCP service requires a *DHCP server* entry in the LDAP directory. The entry does not normally need to be created manually, instead it is created by the join script of the **`univention-dhcp`** package. However, to create another record manually, a DHCP service must be selected in the left column of the UMC module *DHCP*. *Add* ▸ *DHCP Server* can then be used to register a new server.

Table 11.14: *General* tab

Attribute	Description
Server name	The computer name that the DHCP service should offer is entered in this input field, e.g., `ucs-primary`. A server can only ever provide a single DHCP service and therefore cannot be entered in more than one DHCP service at the same time.

Administration of DHCP subnets

DHCP subnets are managed in the UMC module *DHCP* (see *Univention Management Console modules* (page 69)). To create another subnet, a DHCP service must be selected in the left column. *Add* ▸ *DHCP: Subnet* can be used to create a new subnet.

A DHCP subnet entry is required for every subnet from which dynamic or fixed IP addresses are to be assigned. It is only necessary to enter IP address ranges if IP addresses are to be assigned dynamically.

If *DHCP shared subnet* objects are to be used, the corresponding subnets should be created below the *DHCP shared subnet* container created for this purpose (see *Management of DHCP shared networks / DHCP shared subnets* (page 224)).

Table 11.15: *General* tab

Attribute	Description
Subnet address	The IP address of the subnet must be entered in dot-decimal form in this input field, e.g., `192.0.2.0`.
Net mask	The network mask can be entered in this input field as the network prefix or in dot-decimal form. If the network mask is entered in dot-decimal form it will be subsequently be converted into the corresponding network prefix and later also shown so.
Dynamic address assignment	Here one can set up individual or multiple IP address ranges for dynamic assignment. The range stretches from the *First address* to the *Last address* in dot-decimal form. **Caution:** Dynamic IP ranges for a subnet should always either be specified exclusively in the subnet entry or exclusively in one or more special pool entries. The types of IP range entries within a subnet must not be mixed! If different IP ranges with different configurations are be set up in one subnet, pool entries must be created for this purpose.

At this level, the gateway for all computers in a subnet is often set using the *Policy: DHCP Routing* tab (unless other entries are performed at lower levels).

Administration of DHCP pools

DHCP pools can only be managed via the UMC module *LDAP directory*. To do so, one must always be in a DHCP subnet object - a DHCP pool object must always be created below a DHCP subnet object - and a *DHCP: Pool* object added with *Add*.

If DHCP pools are created in a subnet, no IP address range should be defined in the subnet entry. These should only be specified via the pool entries.

General tab

Table 11.16: *General* tab

Attribute	Description
Name	An unambiguous name for the DHCP pool must be entered in this input field, e.g., `testnet.compaby.example`.
Dynamic range	Here you can enter the IP addresses in dot-decimal form that are to be dynamically assigned.

Advanced settings tab

Table 11.17: *Advanced settings* tab

Attribute	Description
Failover peer	The name of a failover configuration, which must to be configured manually in file `/etc/dhcp/local.conf`. Further information can be found at A Basic Guide to Configuring DHCP Failover[49].
Allow known clients	A computer is identified by its MAC address. If this input field is set to `allow` or unset, a computer **with** a matching DHCP host entry (see *Registration of computers with DHCP computer objects* (page 223)) is eligible to receive an IP address from this pool. If set to `deny`, the computer doesn't receive an IP address from the pool.
Allow unknown clients	A computer is identified by its MAC address. If this input field is set to `allow` or unset, a computer **without** a matching DHCP host entry (see *Registration of computers with DHCP computer objects* (page 223)) is eligible to receive an IP address from this pool. If set to `deny`, the computer doesn't receive an IP address from the pool.
Allow dynamic BOOTP clients	BOOTP is the predecessor of the DHCP protocol. It has no mechanism to renew leases and by default assigns leases infinitely, which can deplete the pool. If this options is set to `allow` clients can retrieve an IP address from this pool using BOOTP.
All clients	If this option is set to `deny` the pool is disabled globally. This is only useful in exceptional scenarios.

Registration of computers with DHCP computer objects

A *DHCP host* entry is used to register the respective computer in the DHCP service. Computers can be handled depending on their registration status. Known computers may get fixed and dynamic IP addresses from the DHCP service; unknown computers only get dynamic IP addresses.

DHCP computer entries are usually created automatically when a computer is added via the computer management. Below the DHCP service object you have the possibility of adding DHCP computer entries or editing existing entries manually, irrespective of whether they were created manually or automatically.

DHCP host objects are managed in the UMC module *DHCP* (see *Univention Management Console modules* (page 69)). To register a host in the DHCP manually, a DHCP service must be selected in the left column of the module. *Add ▸ DHCP: Host* can be used to register a host.

Table 11.18: *General* tab

Attribute	Description
Hostname	A name for the computer is entered in this input field (which usually also has an entry in the computer management). It is recommended to enter the same name and the same MAC address for the computer in both entries to facilitate assignment.
Type	The type of network used can be selected in this selection list. *Ethernet* almost always needs to be selected here.
Address	The MAC address of the network card needs to be entered here, e.g., `2e:44:56:3f:12:32` or `2e-44-56-3f-12-32`.
Fixed IP addresses	One or more fixed IP addresses can be assigned to the computer here. In addition to an IP address, a fully qualified domain name can also be entered, which is resolved into one or more IP addresses by the DHCP server.

[49] https://kb.isc.org/docs/aa-00502

Management of DHCP shared networks / DHCP shared subnets

DHCP shared network objects accept subnets which use a common physical network.

DHCP shared network objects are managed in the UMC module *DHCP* (see *Univention Management Console modules* (page 69)). To create a shared network, a DHCP service must be selected in the left column of the module. *Add* ▸ *DHCP: Shared Network* can be used to register a network.

Caution: A shared network must contain at least one shared subnet object. Otherwise the DHCP service will terminate itself and cannot be restarted until the configuration is fixed.

Table 11.19: *General* tab

Attribute	Description
Shared network name	A name for the shared network must be entered in this input field.

Subnets are declared as a *DHCP shared subnet* when they use the same, common physical network. All subnets which use the same network must be stored below the same shared network container. A separate *DHCP shared subnet* object must be created for each subnet.

DHCP shared subnet objects can only be managed via the UMC module *LDAP directory*. To do so, one must always be in a DHCP shared network object - a DHCP shared subnet object must always be created below a DHCP shared network object - and a *DHCP shared subnet* object added with *Add*.

11.3.2 Configuration of clients via DHCP policies

Note: Many of the settings for DHCP are configured via policies. They are also applied to DHCP computer objects if a policy is linked to the LDAP base or one of the other intermediate containers. As the settings for DHCP computer objects have the highest priority, other settings for subnetwork and service objects are ignored.

For this reason, DHCP policies should be linked directly to the DHCP network objects (e.g., the DHCP subnetworks).

Alternatively, the LDAP class `univentionDhcpHost` can be added in the advanced settings of the policies under *Object* ▸ *Excluded object classes*. Such policies are then no longer applied to the DHCP computer objects, with the result that the settings from the DHCP subnetwork and service are used.

Tip: When using the command line **`udm dhcp/host list`** (see also *DNS/DHCP* (page 87)), it is possible to use the option `--policies 0` to display the effective settings.

Setting the gateway

The default gateway can be specified via DHCP with a *DHCP routing* policy, which is managed in the UMC module *Policies* (see *Policies* (page 74))

Table 11.20: *General* tab

Attribute	Description
Routers	The names or IP addresses of the routers are to be entered here. It must be verified that the DHCP server can resolve these names in IP addresses. The routers are contacted by the client in the order in which they stand in the selection list.

Setting the DNS servers

The name servers to be used by a client can be specified via DHCP with a *DHCP DNS* policy, which is managed in the UMC module *Policies* (see *Policies* (page 74))

Table 11.21: *General* tab

Attribute	Description
Domain name	The name of the domain, which the client automatically appends on computer names that it sends to the DNS server for resolution and which are not FQDNs. Usually this is the name of the domain to which the client belongs.
Domain name servers	Here IP addresses or fully qualified domain names (FQDNs) of DNS servers can be added. When using FQDNs, it must be verified that the DHCP server can resolve the names in IP addresses. The DNS servers are contacted by the clients according to the order specified here.

Setting the WINS server

The WINS server to be used can be specified via DHCP with a *DHCP NetBIOS* policy, which is managed in the UMC module *Policies* (see *Policies* (page 74))

Table 11.22: *General* tab

Attribute	Description
NetBIOS name servers	The names or IP addresses of the NetBIOS name servers (also known as WINS servers) should be entered here. It must be verified that the DHCP server can resolve these names in IP addresses. The servers entered are contacted by the client in the order in which they stand in the selection list.
NetBIOS scope	The NetBIOS over TCP/IP scope for the client according to the specification in **RFC 1001**[50] and **RFC 1002**[51]. Attention must be paid to uppercase and lowercase when entering the NetBIOS scope.
NetBIOS node type	This field sets the node type of the client. Possible values are: • `1 B-node` (Broadcast: no WINS) • `2 P-node` (Peer: only WINS) • `4 M-node` (Mixed: first Broadcast, then WINS) • `8 H-node` (Hybrid: first WINS, then Broadcast)

Configuration of the DHCP lease

The validity of an assigned IP address - a so-called DHCP lease - can be specified with a *DHCP lease time* policy, which is managed in the UMC module *Policies* (see *Policies* (page 74))

Table 11.23: *General* tab

Attribute	Description
Default lease time	If the client does not request a specific lease time, the standard lease time is assigned. If this input field is left empty, the DHCP server's default value is used.
Maximum lease time	The maximum lease time specifies the longest period of time for which a lease can be granted. If this input field is left empty, the DHCP server's default value is used.
Minimum lease time	The minimum lease time specifies the shortest period of time for which a lease can be granted. If this input field is left empty, the DHCP server's default value is used.

[50] https://datatracker.ietf.org/doc/html/rfc1001.html
[51] https://datatracker.ietf.org/doc/html/rfc1002.html

Configuration of boot server/PXE settings

A *DHCP Boot* policy is used to assign computer configuration parameters for booting via BOOTP/PXE. They are managed in the UMC module *Policies* (see *Policies* (page 74))

Table 11.24: *Boot* tab

Attribute	Description
Boot server	The IP address or the FQDN of the PXE boot server from which the client should load the boot file is entered in the input field. If no value is entered in this input field, the client boots from the DHCP server from which it retrieves its IP address.
Boot filename	The path to the boot file is entered here. The path must be entered relative to the base directory of the TFTP service (`/var/lib/univention-client-boot/`).

Further DHCP policies

There are some further DHCP policies available, but they are only required in special cases.

DHCP Dynamic DNS
DHCP Dynamic DNS allows the configuration of dynamic DNS updates. These cannot yet be performed with a LDAP-based DNS service as provided out-of-the-box by UCS.

DHCP Allow/Deny
DHCP Allow/Deny allows the configuration of different DHCP options, which control what clients are allowed to do. The are only useful in exceptional cases.

DHCP statements
DHCP statements allows the configuration of different options, which are only required in exceptional cases.

11.4 Packet filter with Univention Firewall

Univention Firewall integrates a packet filter based on **`iptables`** in Univention Corporate Server.

It permits targeted filtering of undesired services and the protection of computers during installations. Furthermore it provides the basis for complex scenarios such as firewalls and application level gateways. Univention Firewall is included in all UCS installations as standard.

By default all incoming ports are blocked. Every UCS package provides rules, which free up the ports required by the package again.

The configuration is primarily performed via Univention Configuration Registry variables. The definition of this type of packet filter rules is documented in *Univention Developer Reference* [3].

In addition, the configuration scripts in the `/etc/security/packetfilter.d/` directory are listed in alphabetic order. The names of all scripts begin with two digits, which allows a numbered order. The scripts must be marked as executable.

After changing the packet filter settings, the **`univention-firewall`** service has to be restarted.

Univention Firewall can be deactivated by setting the Univention Configuration Registry Variable *`security/packetfilter/disabled`* (page 304) to `true`

11.5 Web proxy for caching and policy management / virus scan

The UCS proxy integration allows the use of a web cache for improving the performance and controlling data traffic. It is based on the tried-and-tested proxy server Squid and supports the protocols HTTP, FTP and HTTPS.

A proxy server receives requests about internet contents and verifies whether these contents are already available in a local cache. If this is the case, the requested data are provided from the local cache. If the data are not available, these contents are called up from the respective web server and inserted in the local cache. This can be used to reduce the answering times for the users and the transfer volume via the internet access.

Further documentation on proxy services - such as the cascading of proxy servers, transparent proxies and the integration of a virus scan engine - are documented in *Extended IP and network management documentation* [14].

11.5.1 Installation

Squid can be installed from the Univention App Center with the application **Web proxy / web cache (Squid)**. Alternatively, the software package **univention-squid** can be installed. Additional information can be found in *Installation of further software* (page 99).

The service is configured with standard settings sufficient for operation so that it can be used immediately. It is possible to configure the port on which the service is accessible to suit your preferences (see *Access port* (page 228)); port `3128` is set as default.

If changes are made to the configuration, Squid must be restarted. This can be performed either via the UMC module *System services* or the command line:

```
$ systemctl restart squid
```

In addition to the configuration possibilities via Univention Configuration Registry described in this document, it is also possible to set additional Squid configuration options in the `/etc/squid/local.conf`.

11.5.2 Caching of web content

Squid is a caching proxy, i.e., previously viewed contents can be provided from a cache without being reloaded from the respective web server. This reduces the incoming traffic via the internet connection and can result in quicker responses of HTTP requests.

However, this caching function is not necessary for some environments or, in the case of cascaded proxies, it should not be activated for all of them. For these scenarios, the caching function of the Squid can be deactivated with the Univention Configuration Registry Variable *squid/cache* (page 305) by setting this to `no`. Squid must then be restarted.

11.5.3 Logging proxy accesses

All accesses performed via the proxy server are stored in the log file `/var/log/squid/access.log`. It can be used to follow which websites have been accessed by the users.

11.5.4 Restriction of access to permitted networks

As standard, the proxy server can only be accessed from local networks. If, for example, a network interface with the address `192.0.2.10` and the network mask `255.255.255.0` is available on the computer on which Squid is installed, only computers from the network `192.0.2.0/24` can access the proxy server. Additional networks can be specified via the Univention Configuration Registry Variable *squid/allowfrom* (page 304). When doing so, the CIDR notation must be used; several networks should be separated by blank spaces.

Example:

```
$ univention-config-registry set squid/allowfrom="192.0.2.0/24 192.0.3.0/24"
```

Once Squid has been restarted, access is now permitted from the networks `192.0.2.0/24` and `192.0.3.0/24`. If configured to `all`, proxy access in granted from all networks.

11.5.5 Configuration of the ports used

Access port

As standard, the web proxy can be accessed via port `3128`. If another port is required, this can be configured via the Univention Configuration Registry Variable *squid/httpport* (page 305). If Univention Firewall is used, the packet filter configuration must also be adjusted.

Permitted ports

In the standard configuration, Squid only forwards client requests intended for the network ports 80 (HTTP), 443 (HTTPS) or 21 (FTP). The list of permitted ports can be changed via the Univention Configuration Registry Variable *squid/webports* (page 305); several entries should be separated by blank spaces.

Example:

```
$ univention-config-registry set squid/webports="80 443"
```

With this setting, access is only allowed to ports 80 and 443 (HTTP and HTTPS).

11.5.6 User authentication on the proxy

It is sometimes necessary to restrict web access to certain users. Squid allows user-specific access regulation via group memberships. To allow verification of group membership, it is necessary for the user to authenticate on the proxy server.

Caution: To prevent unauthorized users from opening websites nonetheless, additional measures are required to prevent these users from bypassing the proxy server and accessing the internet. This can be done, for example, by limiting all HTTP traffic through a firewall.

The proxy authentication (and as a result the possible verification of the group memberships) must firstly be enabled. There are three possible mechanisms for this:

LDAP server authentication
: Direct authentication against the LDAP server. This is done by setting the Univention Configuration Registry Variable *squid/basicauth* (page 305) to `yes` and restarting Squid.

NTLM authentication
: Authentication is performed via the NTLM interface. Users logged in on a Windows client then do not need to authenticate themselves again when accessing the proxy. NTLM authentication is enabled by setting the Univention Configuration Registry Variable *squid/ntlmauth* (page 305) to `yes` and restarting Squid.

Kerberos authentication

Authentication is performed via Kerberos. Users logged in on a Windows client which is a member of a Samba/AD domain authenticate themselves on the proxy with the ticket that they received when they logged in to the domain. The **univention-squid-kerberos** package must be installed on every proxy server for it to be possible to enable Kerberos authentication. Then the Univention Configuration Registry Variable *squid/krb5auth* (page 305) must be set to `yes` and Squid restarted.

If NTLM is used an NTLM authentication is performed for every HTTP query as standard. If for example the website <https://www.univention.com/> is opened, the subsequent pages and images are loaded in addition to the actual HTML page. The NTLM authentication can be cached per domain: If the Univention Configuration Registry Variable *squid/ntlmauth/keepalive* (page 305) is set to `yes`, no further NTLM authentication is performed for subsequent HTML queries in the same domain. In case of problems with local user accounts it may help to set this variable to `no`.

In the standard setting all users can access the proxy. The Univention Configuration Registry Variable *squid/auth/allowed_groups* (page 304) can be used to limit the proxy access to one or several groups. If several groups are specified, they must be separated by a semicolon.

11.6 RADIUS

The **RADIUS** app increases the security for UCS managed IT infrastructures by controlling the access to the wireless network for users, groups and endpoint devices via RADIUS protocol[52]. The configuration is done via deny and allow lists and directly at user, group and endpoint device objects in the UCS management system. Registered users are authenticated with their usual domain credentials or, alternatively, with a specifically for RADIUS generated password, which, among others, also allows bring your own device concepts.

11.6.1 Installation

RADIUS is available through the App Center (see *Univention App Center* (page 92)) and can be installed using the UMC module *App Center*. It can be installed on multiple machines. After the installation it runs a FreeRADIUS[53] server. Authenticators (e.g. access points) can contact via RADIUS to check network access requests.

The RADIUS app can also be installed on UCS@school systems. In this case, the network access can be given to users or groups regardless of the internet rule or computer room settings.

11.6.2 Configuration

Allowed users

By default no user is allowed to access the network. Enabling the checkbox for *network access* on the *RADIUS* tab, gives the user access to the network. The checkbox can also be set on groups, which allows all users in this group access.

[52] https://en.wikipedia.org/wiki/RADIUS
[53] https://www.freeradius.org/

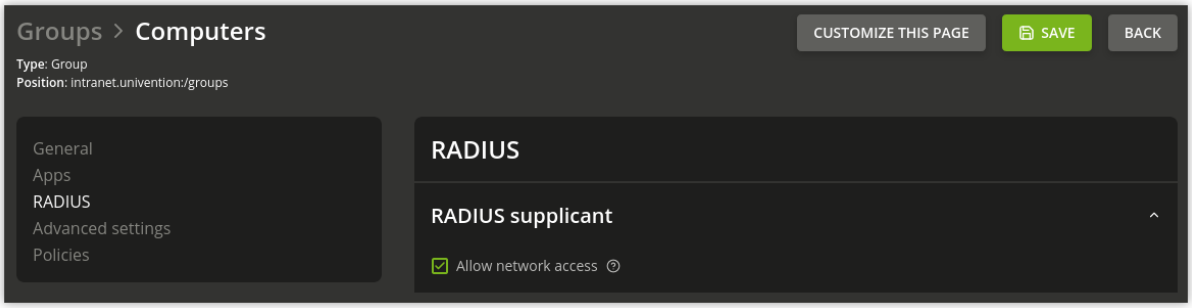


Fig. 11.4: Example for a group allowing network access to its users

Service specific password

By default, users authenticate with their domain password. By setting the Univention Configuration Registry Variable *radius/use-service-specific-password* (page 303) to `true`, a dedicated password for RADIUS will be used. Through the *Self Service app* (page 119), users can get such a password. The system will generate a random password for users to use. If needed, a new password can be generated at any time. This also invalidates the old password. To enable this page in the Self Service, the Univention Configuration Registry Variable *umc/self-service/service-specific-passwords/backend/enabled* (page 119) has to be set to `true` on the *Self Service Backend*.

The parameters used to generate the passwords can be adjusted. On a Primary Directory Node some Univention Configuration Registry Variables have to be set:

```
$ ucr search password/radius/quality
```

MAC filtering

By default access to the network is allowed for every device (assuming the used username has access). It can be restricted to only allow specific devices. This can be enabled by setting the Univention Configuration Registry Variable *radius/mac/whitelisting* (page 303) to `true`. When enabled, the device used to access the network is looked up via the LDAP attribute `macAddress` and the resulting computer object must have network access granted (either directly or via one of its groups), too.

MAC Authentication Bypass with computer objects

MAC Authentication Bypass (MAB) is a proprietary fallback mode to 802.1X for devices that don't support 802.1X authentication, such as network printers or wireless phones. MAB is an option that allows such devices to authenticate with the network using their MAC address as their username.

This section describes how to use a device's MAC address for authentication and assign them a VLAN to the corresponding network infrastructure through MAB. To activate MAC Authentication Bypass, set the Univention Configuration Registry Variable *freeradius/conf/allow-mac-address-authentication* (page 293) to `true`.

Important: Devices that authenticate using MAB ignore network access settings:

- Univention Configuration Registry Variable *radius/mac/whitelisting* (page 303)
- The checkbox *Allow network access* at the computer object and in the group setting

---

Warning: Attackers can spoof MAC addresses. Consider any port as compromised where your switch allows to use MAB. Make sure you have put appropriate measures in place to still keep your network secure.

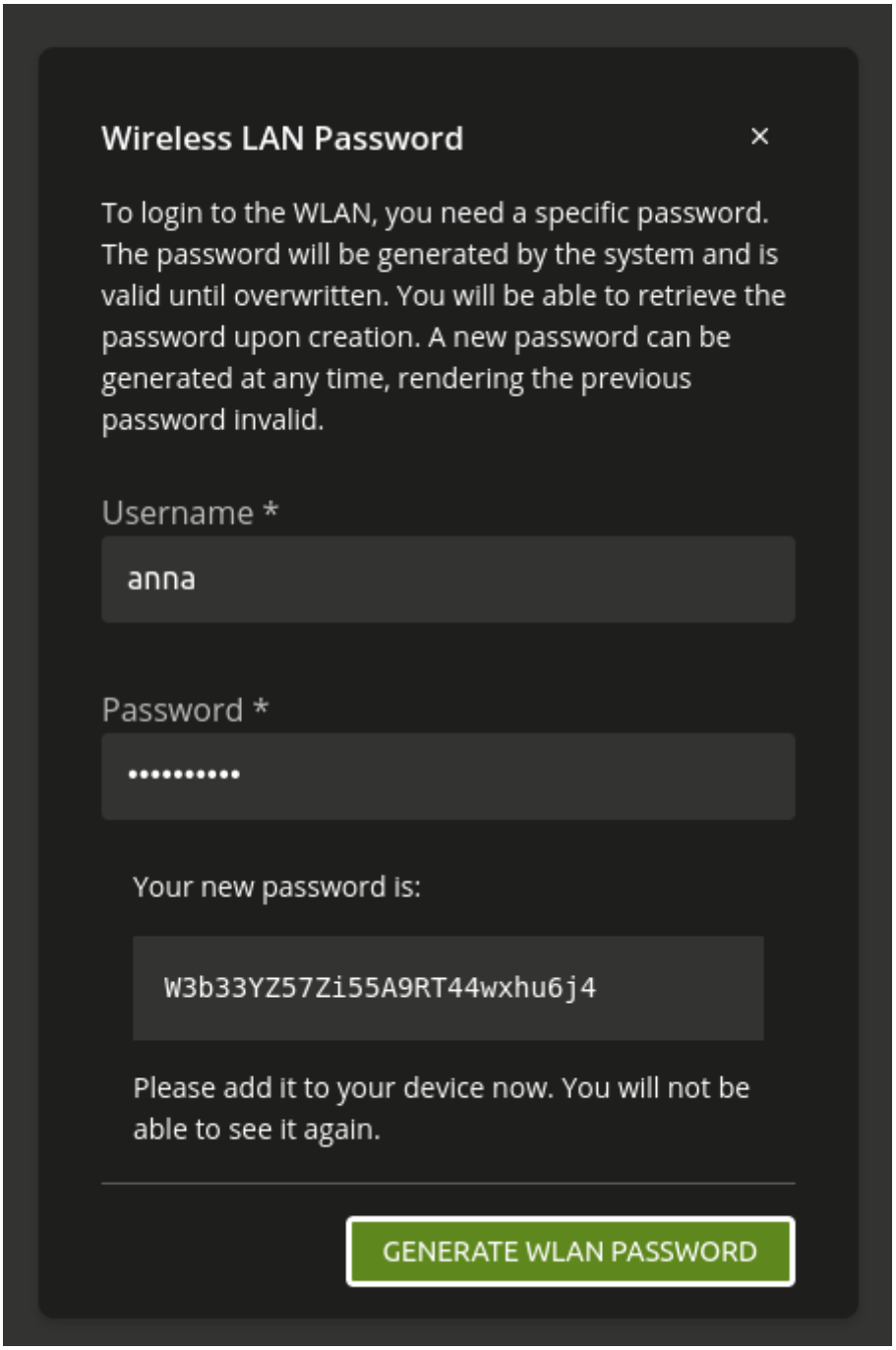


Fig. 11.5: The page in the Self Service to get a RADIUS specific password

To assign the VLAN ID to a computer, you need to add it to the group of the computer object with the respective VLAN ID. In the UCS management system, follow these steps:

1. Open *Devices* ‣ *Computers*.
2. Click the computer object to edit.
3. Go to *Advanced settings* ‣ *Groups*.
4. To add a group with VLAN IDs, click + *ADD*, select `Virtual LAN ID` from the *Object property* drop-down, and activate the appropriate group to add it.
5. To save, click *ADD* in the *Add objects* dialog and *SAVE* in the *Advanced settings*.

To assign the VLAN ID to a user group, you need to add it to the user group settings. In the UCS management system, follow these steps:

1. Open *Users* ‣ *Groups*.
2. Click the user group object to edit or create a new user group.
3. Go to *RADIUS*.
4. Enter the VLAN ID as number into the field *Virtual LAN ID*.
5. To save, click *SAVE*.

If a computer object has assigned several groups with VLAN IDs, UCS selects the VLAN ID with the lowest number and assigns it. To configure a default VLAN ID, set it as value to the Univention Configuration Registry Variable *freeradius/vlan-id* (page 294).

After you completed the configuration, the Radius server returns the assigned VLAN ID to requests with the given MAC address.

UCS stores the MAC address in the LDAP directory as lowercase string with the colon (`:`) as separator, for example `00:00:5e:00:53:00`.

New in version 5.0-6-erratum-1011: With UCS 5.0 erratum 1011[54] the Radius server can handle different formats of the MAC addresses for the username when using MAB.

Devices that use MAB, use their MAC address as username and they may use different formats for it. The Radius server supports different case-sensitive formats. The following list shows the tested formats:

- `XX:XX:XX:XX:XX:XX`
- `XX-XX-XX-XX-XX-XX`
- `XX.XX.XX.XX.XX.XX`
- `XXXX.XXXX.XXXX`
- `XXXXXXXXXXXX`

Note: For non-standard formats, you can configure a regular expression in the Univention Configuration Registry Variable *freeradius/conf/mac-addr-regexp* (page 293) to match your custom MAC address format.

Depending on your regular expression, the previously listed formats may not work anymore.

Important: All devices that use MAB, need to have the same password set, because *service specific passwords* (page 230) don't work, and the switch must know the password. You can only configure one device password in the switch. You can make up your own password for the devices using MAB, for example `mab request format attribute 2 password1`.

[54] https://errata.software-univention.de/#/?erratum=5.0x1011

If the network infrastructure provides a different format, you can often reconfigure the format. For example, for Cisco switches, you can use `mab request format attribute 1 groupsize 2 separator : lowercase` as described in Configurable MAB Username and Password[55].

Access point administration

All access points must be known to the RADIUS server. An access point can either be configured in the file `/etc/freeradius/3.0/clients.conf` or through the UMC module *Computers*. For each access point a random shared secret should be created (e.g. by using the command **`makepasswd`**). The `shortname` can be chosen at will.

Example entry for an access point:

```
client AP01 {
    secret = a9RPAeVG
    ipaddr = 192.0.2.101
}
```

To configure an access point using the UMC module *Computers* create or select a computer object and activate the *RADIUS-Authenticator* option (*RADIUS option* (page 233)). An *IP client* is a good choice as a computer object for access points. The RADIUS settings can be edited on the *RADIUS* tab of the object (*RADIUS authenticator options* (page 234)). At least the IP address and the shared secret must be configured. The virtual server and NAS type options usually do not need to be changed.

Access points that are configured via the UMC module *Computers* are available to all RADIUS servers in the domain. To achieve this, the Univention Directory Listener will write them into the file `/etc/freeradius/3.0/clients.univention.conf` and restart the RADIUS server. In order to merge multiple changes in one restart, this happens with a slight delay (around 15 seconds). New access points can only access the RADIUS server after this restart.

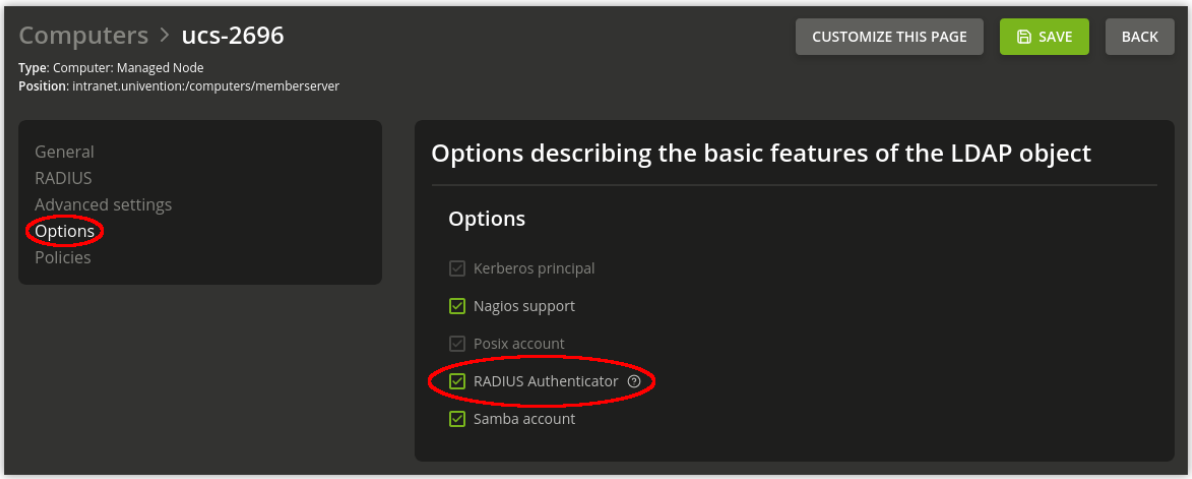


Fig. 11.6: RADIUS option

[55] https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_usr_aaa/configuration/15-e/sec-usr-aaa-15-e-book/sec-usr-config-mab-usrname-pwd.html

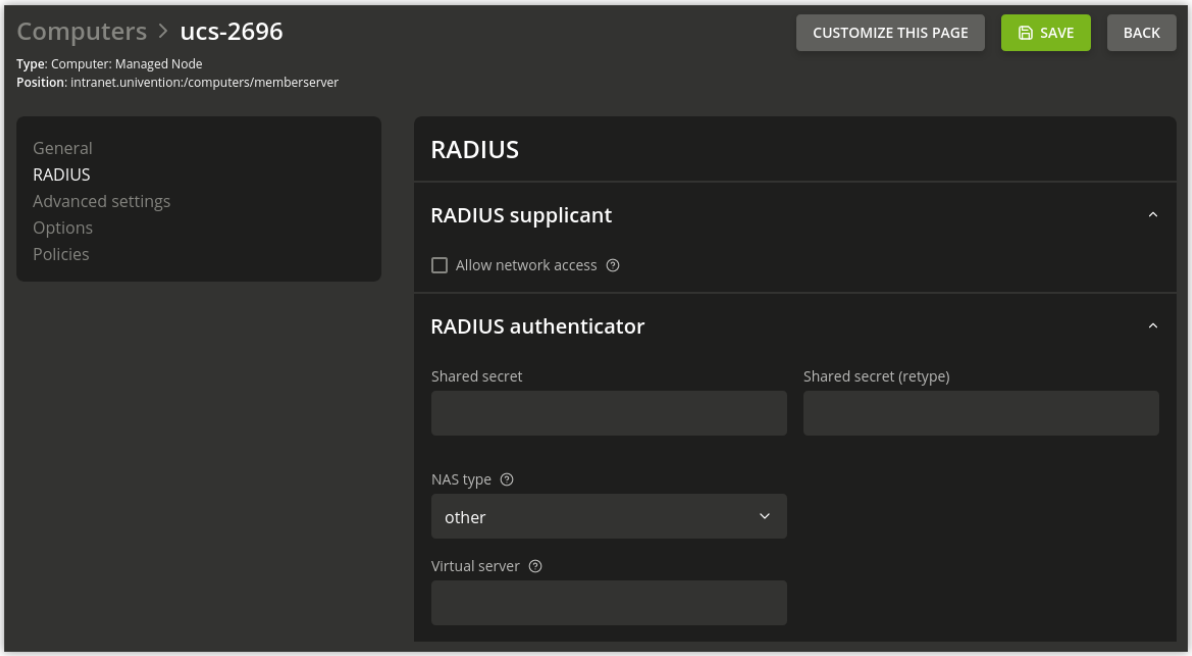


Fig. 11.7: RADIUS authenticator options

Access point and client configuration

The access points must then be configured to use 802.1x ("WPA Enterprise") authentication. And the *RADIUS server* address should be set to the address of the server, where the RADIUS app is installed. The password must be set to the `secret` from the `clients.conf` entry for that access point.

Wireless clients have to be configured to use *WPA* with *PEAP* and *MSCHAPv2* for authentication.

VLAN IDs

Virtual Local Area Networks (VLANs) can be used to separate the traffic of users at the network level. UCS can be configured to return a VLAN ID in the Radius response of the Radius authentication process according to **RFC 3580 / IEEE 802.1X**[56]. You find further information in *Configure VLAN* (page 154).

The VLAN ID for a user can be configured by assigning the user to a group with a VLAN ID.

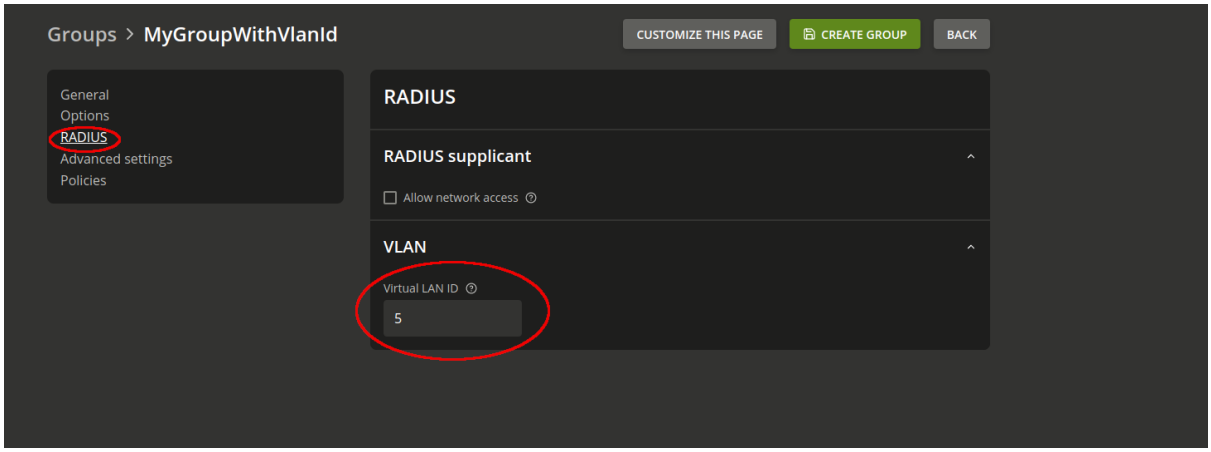


Fig. 11.8: Assigning VLAN ID to a user group

[56] https://datatracker.ietf.org/doc/html/rfc3580.html

A default VLAN ID can be configured in the Univention Configuration Registry Variable *freeradius/vlan-id* (page 294). This default VLAN ID will be returned if the user is not a member of a group with a VLAN ID. The Radius response will not contain any VLAN ID in case the user is not a member of a group with VLAN ID and no default VLAN ID is defined.

11.6.3 Debugging

The **RADIUS** app has a log file under `/var/log/univention/radius_ntlm_auth.log`. The log verbosity can the controlled via the Univention Configuration Registry Variable *freeradius/auth/helper/ntlm/debug* (page 293). The **FreeRADIUS server** uses the log file: `/var/log/freeradius/radius.log`.

The tool **univention-radius-check-access** can be used to evaluate the current access policy for a given user and/or station ID (MAC address). It can be executed as root on the server where **univention-radius** its installed:

```
root@primary211:~# univention-radius-check-access --username=stefan
DENY 'uid=stefan,cn=users,dc=ucs,dc=example'
'uid=stefan,cn=users,dc=ucs,dc=example'
-> DENY 'cn=Domain Users,cn=groups,dc=ucs,dc=example'
-> 'cn=Domain Users,cn=groups,dc=ucs,dc=example'
-> -> DENY 'cn=Users,cn=Builtin,dc=ucs,dc=example'
-> -> 'cn=Users,cn=Builtin,dc=ucs,dc=example'
Thus access is DENIED.
```

```
root@primary211:~# univention-radius-check-access --username=janek
DENY 'uid=janek,cn=users,dc=ucs,dc=example'
'uid=janek,cn=users,dc=ucs,dc=example'
-> DENY 'cn=Domain Users,cn=groups,dc=ucs,dc=example'
-> ALLOW 'cn=Network Access,cn=groups,dc=ucs,dc=example'
-> 'cn=Domain Users,cn=groups,dc=ucs,dc=example'
-> -> DENY 'cn=Users,cn=Builtin,dc=ucs,dc=example'
-> -> 'cn=Users,cn=Builtin,dc=ucs,dc=example'
-> 'cn=Network Access,cn=groups,dc=ucs,dc=example'
Thus access is ALLOWED.
root@primary211:~#
```

It prints a detailed explanation and sets the exit code depending on the result of the access check (`0` for *access granted*, `1` for *access denied*).

CHAPTER

TWELVE

FILE SHARE MANAGEMENT

UCS supports the central management of directory shares. A share registered via the UMC module *Shares* is created on an arbitrary UCS server system as part of the UCS domain replication.

Provision for accessing clients can occur via CIFS (supported by Windows/Linux clients) and/or NFS (primarily supported by Linux/Unix). The NFS shares managed in the UMC module can be mounted by clients both via NFSv3 and via NFSv4.

If a file share is deleted on a server, the shared files in the directory are preserved.

To be able to use access control lists on a share, the underlying Linux file system must support POSIX ACLs. In UCS the file systems `ext4` and `XFS` support POSIX ACLs. The Samba configuration also allows storing DOS file attributes in extended attributes of the Unix file system. To use extended attributes, the partition must be mounted using the mount option `user_xattr`.

12.1 Access rights to data in shares

Access permissions to files are managed in UCS using users and groups. All the file servers in the UCS domain access identical user and group data via the LDAP directory.

Three access rights are differentiated per file:

- read
- write
- execute

Three access rights also apply per directory: read and write are the same; the execute permission here refers to the permission to enter a directory.

Each file/directory is owned by a user and a group. The three permission outlined above can be applied to the user owner, the owner group and all others.

setuid
: If the *setuid* option is set for an executable file, it can be run by users with the privileges of the owner of the file.

setgid
: If the *setgid* option is set for a directory, files saved there inherit the directory's owner group. If further directories are created, they also inherit the option.

sticky bit
: If the *sticky bit* option is enabled for a directory, files in this directory can only be deleted by the owner of the file or the root user.

Access control lists allow even more complex permission models. The configuration of ACLs is described in SDB 1042[57].

[57] https://help.univention.com/c/knowledge-base/supported/48

In the Unix permission model - and thus under UCS - write permission is not sufficient to change the permissions of a file. This is limited to the owner/owner group of a file. In contrast, under Microsoft Windows all users with write permissions also have the permission to change the permissions. This scheme can be adjusted for CIFS shares (see *Management of shares via UMC module* (page 238)).

Only initial users and access permissions are assigned when a directory share is created. If the directory already exists, the permissions of the existing directory are adjusted.

Changes to the permissions of a shared directory performed directly in the file system are not forwarded to the LDAP directory. If the permissions/owners are edited with the UMC module *Shares*, the changes in the file system are overwritten. Settings to the root directory of a file share should thus only be set and edited with the UMC module. Additional adjustment of the access permissions of the subordinate directories are then performed via the accessing clients, e.g., via Windows Explorer, or directly via command line commands on the file server.

The *homes* share plays a special role within Samba. This share is used for sharing the home directories of the users. This share is automatically converted to the user's home directory. Samba therefore ignores the rights assigned to the share, and uses the rights of the respective home directory instead.

12.2 Management of shares via UMC module

File shares are managed in the UMC module *Shares* (see *Univention Management Console modules* (page 69)).

When adding/editing/deleting a share, it is entered, modified or removed in the `/etc/exports` file and/or the Samba configuration.

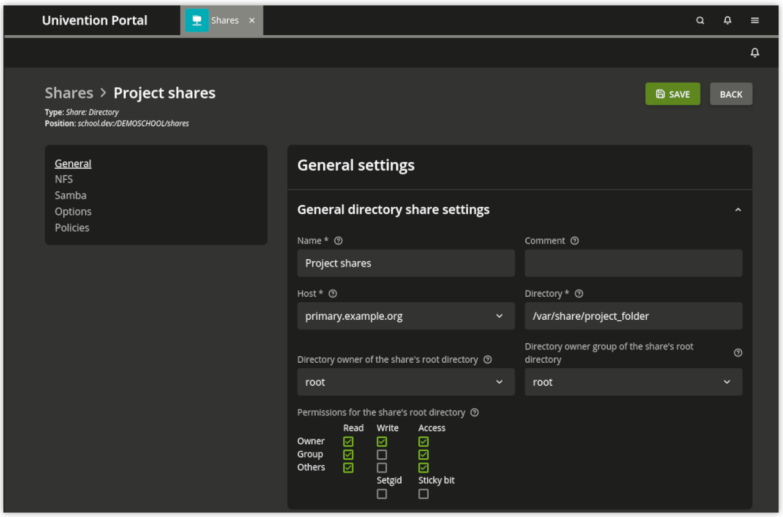


Fig. 12.1: Creating a share via the UMC module *Shares*

12.2.1 Shares UMC module - General tab

Table 12.1: *General* tab

Attribute	Description
Name	The name of the share is to be entered here. The name must be composed of letters, numerals, full stops or blank spaces and must begin and end with a letter or numeral.
Comment	A free selectable description for this share. This is also displayed in the file browser in Windows.
Host	The server where the share is located. All of the Primary/Backup/Replica Directory Node computers and Managed Nodes entered in the LDAP directory for the domain are available for selection which are entered in a DNS forward lookup zone in the LDAP directory.
Directory	The absolute path of the directory to be shared, without quotation marks (this also applies if the name includes special characters such as spaces). If the directory does not exist, it will be created automatically on the selected server. If the Univention Configuration Registry Variable `listener/shares/rename` (page 297) is set to `yes`, the contents of the existing directory are moved when the path is modified. No shares can be created in and below `/proc`, `/tmp`, `/root`, `/dev` and `/sys` and no files can be moved there.
Directory owner of the share's root	The user to whom the root directory of the share should belong, see *Access rights to data in shares* (page 237).
Directory owner group of the share's root	The group to whom the root directory of the share should belong, see *Access rights to data in shares* (page 237).
Permissions for the share's root	The read, write and access permissions for the root directory of the share, see *Access rights to data in shares* (page 237).

12.2.2 Shares UMC module - NFS tab

Shares UMC module - NFS group

Table 12.2: *NFS* group

Attribute	Description
NFS write access	Allows NFS write access to this share; otherwise the share can only be used in read-only mode.
Subtree checking	If only one subdirectory of a file system is exported, the NFS server has to check whether an accessed file is located on the exported file system and in the exported path, each time access is made. Path information is passed on to the client for this check. Activating this function might cause problems if a file opened by the client, is renamed.
Modify user ID for root user (root squashing)	In the NFS standard procedure, identification of users is achieved via user IDs. To prevent a local root user from working with root permissions on other shares, root access can be redirected. If this option is activated, access operations are executed as user `nobody`. The local group `staff`, which is by default empty, owns privileges which come quite close to `root` permissions, yet this group is not considered by the redirection mechanism. This fact should be borne in mind when adding users to this group.
NFS synchronization	The synchronization mode for the share. The `sync` setting is used to write data directly on the underlying storage device. The opposite setting - `async` - can improve performance but also involves the risk of data loss if the server is shut down incorrectly.
Only allow access for these hosts, IP addresses or networks	By default, all hosts are permitted access to a share. In this selection list, host names and IP addresses can be included, to which the access to the share is to be restricted. For example, access to a share containing mail data could be restricted to the mail server of the domain.

Shares UMC module - NFS custom settings group

Table 12.3: *NFS custom settings* group

Attribute	Description
Custom NFS share settings	Apart from the properties in the *NFS* group, this setting makes it possible to define further arbitrary NFS settings for the share. A list of available options can be obtained by the command **`man 5 exports`**. Double entries of configuration options are not checked.

Caution: The definition of extended NFS settings is only necessary in special cases. The options should be thoroughly checked since they might have security-relevant effects.

12.2.3 Shares UMC module - Samba tab

Shares UMC module - Samba group

Table 12.4: *Samba* tab

Attribute	Description
Windows name	The NetBIOS name of the share. This is the name under which the share is displayed on Windows computers in the network environment. When adding a directory share, the UMC module adopts the name entered in the *Name* field of the *General* tab as the default.
Show share in Windows network environment	Specifies whether the share in question is to show up on Windows clients within the network environment.
Allow anonymous read-only access with a guest user	Permits access to this share without a password. Every access is carried out by means of the common guest user `nobody`.
Export share as MSDFS root	This option is documented in *Support for MSDFS* (page 245).
Hide unreadable files/directories	If this option is activated, all files which are not readable for the user due to their file permissions, will be hidden.

Shares UMC module - Samba permissions group

Table 12.5: *Samba permissions* group

Attribute	Description
Users with write access may modify permissions	If this option is activated, all users with write permission to a file are allowed to change permissions, ACL entries, and file ownership rights, see *Access rights to data in shares* (page 237).
Force user	This username and its permissions and primary group is used for performing all the file operations of accessing users. The username is only used once the user has established a connection to the Samba share by using their real username and password. A common username is useful for using data in a shared way, yet improper application might cause security problems.
Force group	A group which is to be used by all users connecting with this share, as their primary group. Thereby, the permissions of this group automatically apply as the group permissions of all these users. A group registered here has a higher priority than a group which was assigned as the primary group of a user via the *Force user* entry field. If a + sign is prefixed to the group name, then the group is assigned as a primary group solely to those users which are already members of this group. All other users retain their primary groups.
Valid users or groups	Names of users or groups which are authorized to access this Samba share. To all other users, access is denied. If the field is empty, all users may access the share - if necessary after entering a password. This option is useful for securing access to a share at file server level beyond the file permissions. The entries are to be separated by spaces. The special characters `@`, `+` and `&` can be used in connection with the group name for assigning certain permissions to the users of the stated group for accessing the Samba share: • A name beginning with the character `@` will first be interpreted as a NIS net-group. Should no NIS net-group of this name be found, the name will be considered as a UNIX group. • A name beginning with the character `+` will be exclusively considered as a UNIX group, a name beginning with the character `&` will be exclusively considered as a NIS net-group. • A name beginning with the characters `+&`, will first be interpreted as a UNIX group. Should no UNIX group of this name be found, the name will be considered as a NIS net-group. The characters `&+` as the beginning of a name correspond to the character `@`.
Invalid users or groups	The users or groups listed here cannot access the Samba share. The syntax is identical to the one for valid users. If a user or group is included in the list of valid users and unauthorized users, access is denied.
Restrict read access to these users/groups	Only the users and groups listed here have read permission for the corresponding share.
Restrict write access to these users/groups	Only the users and groups listed here have write permission for the corresponding share.
Allowed hosts/networks	Names of computers which are authorized to access this Samba share. All other computers are denied access. In addition to computer names, it is also possible to specify IP or network addresses, e.g., `192.0.2.0/255.255.255.0`.
Denied hosts/networks	The opposite to the authorized computers. If a computer appears in both lists, the computer is permitted to access the Samba share.
Inherit ACLs	When activating this option, each file created in this share will inherit the ACL (Access Control List) of the directory where the file was created.
Create files/directories with the owner of the parent directory	When activating this option, each newly created file will not be assigned of the user who created the file, but to the owner of the superior directory instead.
Create files/directories with permissions of the parent directory	When activating this option, for each file or directory created in this share, the UNIX permissions of the superior directory will automatically be adopted.

If a new file is created on a Samba server from a Windows client, the file permissions will be set in several steps:

1. First, only the DOS permissions are translated into UNIX permissions.
2. Then the permissions are filtered via the *Filemode*. UNIX permissions which are marked in *File mode*, are the only ones preserved. Permissions not set here, will be removed. Thus, the permissions have to be set as UNIX permissions and in *File mode* in order to be preserved.
3. In the next step, the permissions under *Force file mode* are added. As a result, the file will have all the permissions set after step 2 or under *Force file mode*. This means, permissions marked under *Force file mode* are set in any case.

Accordingly, a newly created directory will initially be assigned the same permissions as that which are set as UNIX permissions and in *Directory mode* at the same time. Then these permissions are completed by those marked under *Force directory mode*.

Shares UMC module - Samba extended permissions group

Table 12.6: *Samba extended permissions* group

Attribute	Description
File mode	The permissions Samba is to adopt when creating a file, provided they are set under Windows.
Directory mode	The permissions Samba is to adopt when creating a directory, provided they are set under Windows.
Force file mode	The permissions Samba is to set in any case when creating a file, irrespective of whether they are set under Windows or not.
Force directory mode	The permissions Samba is to set in any case when creating a directory, irrespective of whether they are set under Windows or not.

Shares UMC module - Samba options group

Table 12.7: *Samba options* group

Attribute	Description
VFS Objects	Virtual File System (VFS) modules are used in Samba for performing actions before an access to the file system of a share is made, e.g., a virus scanner which stores every infected file accessed in the share in quarantine or server-side implementation of recycle bin deletion of files.
Hidden files	Files and directories to be accessed under Windows, yet not to be visible. Such files or directories are assigned the DOS attribute *hidden*. When entering the names of files and directories, upper and lower case letters are to be differentiated. Each entry is to be separated from the next by a slash. Since the slash can thus not be used for structuring path names, the input of path names is not possible. All files and directories of this name within the share will be hidden. The names may include spaces and the wildcards `*` and `?`. As an example, `/.*/test/` hides all files and directories the names of which begin with a *dot*, or which are called *test*. **Note:** Entries in this field have an impact on the speed of Samba since every time particular contents of the share are to be displayed, all files and directories have to be checked according to the active filters.
Postexec script	A script or command which is to be executed on the server if the connection to this share is finished.
Preexec script	A script or command which is to be executed on the server each time a connection to this share is established.

Shares UMC module - Samba custom settings group

Table 12.8: *Samba custom settings* group

Attribute	Description
Custom share settings	Apart from the properties which can, as a standard feature, be configured in a Samba share, this setting makes it possible to define further arbitrary Samba settings within the share. A list of available options can be obtained by the command **`man smb.conf`**. In *Key* the name of the option is to be entered, and in the *Value* field the value to be set. Double entries of configuration options are not checked.

Caution: The definition of extended Samba settings is only necessary in very special cases. The options should be thoroughly checked since they might have security-relevant effects.

12.2.4 Shares UMC module - Options tab

Table 12.9: *Options* tab

Attribute	Description
Export for Samba clients	This option defines whether the share is to be exported for Samba clients.
Export for NFS clients	This option defines whether the share is to be exported for NFS clients.

12.3 Support for MSDFS

The Microsoft Distributed File System (MSDFS) is a distributed file system which makes it possible to access shares spanning several servers and paths as a virtual directory hierarchy. The load can then be distributed across several servers.

Setting the *MSDFS Root* option for a share (see *Management of shares via UMC module* (page 238)) indicates that the shared directory is a share which can be used for the MSDFS. References to other shares are only displayed in such an MSDFS root, elsewhere they are hidden.

To be able to utilize the functions of a distributed file system, the Univention Configuration Registry Variable `samba/enable-msdfs` (page 303) has to be set to `yes` on a file server. Afterwards Samba has to be restarted.

For creating a reference named `tofb` from server `sa` within the share `fa` to share `fb` on the server `sb`, the following command has to be executed in directory `fa`:

```
$ ln -s msdfs:sb\\fb tofb
```

This reference will be displayed on every client capable of MSDFS (e.g. *Windows 2000* and *Windows XP*) as a regular directory.

Caution: Only restricted user groups should have write access to root directories. Otherwise, it would be possible for users to redirect references to other shares, and intercept or manipulate files. In addition, paths to the shares, as well as the references are to be spelled entirely in lower case. If changes are made in the references, the concerned clients have to be restarted.

Further information on this issue can be found in Jelmer R. Vernooij and Carter [15].

12.4 Configuration of file system quota

UCS allows the limiting of the storage space available to a user on a partition. These thresholds can be set as either a quantity of storage space (e.g., 500 MB per user) or a maximum number of files without a defined size limit.

Two types of thresholds are differentiated:

Hard limit
: The *hard limit* is the maximum storage space a user can employ. If it is attained, no further files can be saved.

Soft limit
: If the *soft limit* is attained - which must be smaller than the hard limit - and the storage space used is still below the hard limit, the user is given a grace period of seven days to delete unused data. Once seven days have elapsed, it is no longer possible to save or change additional files. A warning is displayed to users who access a file system with an exceeded quota via CIFS (the threshold is based on the soft limit).

If a quota value of `0` has been configured, it is evaluated as an unlimited quota.

Quotas can either be defined via the UMC module *Filesystem quotas* or a policy for shares, see *Configuring file system quota* (page 246).

File system quotas can only be applied on partitions with the file systems `ext4` and `XFS`. Before file system quotas can be configured, the use of file system quotas needs to be activated per partition, see *Activating file system quota* (page 246).

12.4.1 Activating file system quota

In the UMC module *Filesystem quotas*, all the partitions are listed on which quotas can be set up. Only partitions are shown which are currently mounted under a mount point.

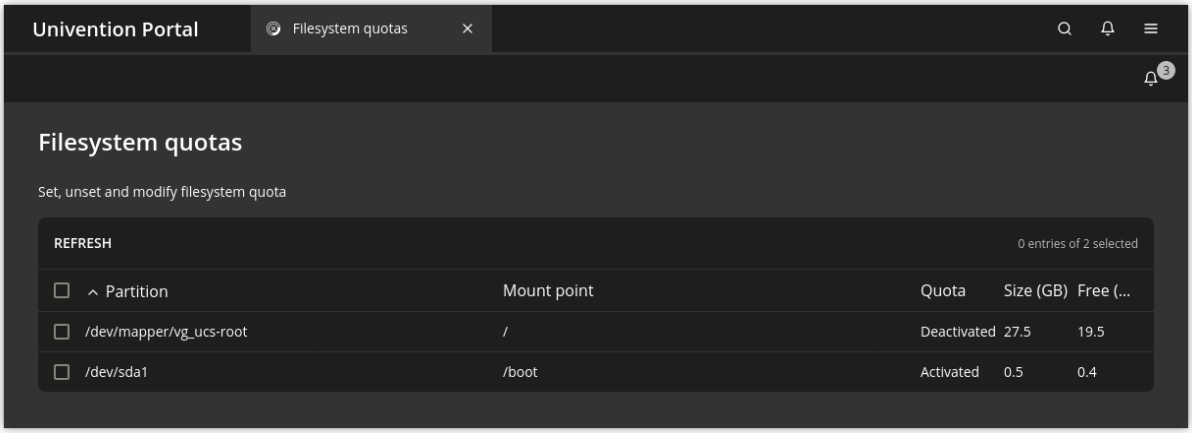


Fig. 12.2: The UMC module *File system quotas*

The current quota status (activated/deactivated) is shown and can be changed with *Activate* and *Deactivate*.

If quota has been activated on a `XFS` root-partition, the system has to be rebooted.

12.4.2 Configuring file system quota

Quotas can either be defined via the UMC module *Filesystem quotas* or a policy for shares, see *Policies* (page 74). The configuration through a policy allows setting a default value for all users, while the UMC module allows configuration of user-specific quota values.

The user-specific quota settings can be configured with the UMC module *Filesystem quotas*. The permitted storage quantities can be set with the pencil symbol for all enabled partitions. All the settings are set user-specifically. *Add* can be used to set the thresholds for soft and hard limits for a user.

The quota settings can also be set with a *User quota* share policy. The settings apply for all users of a share; it is not possible to establish different quota limits for different users within one policy.

Quota settings that are applied via a quota policy are by default only applied once to the file system. If the setting is changed, it will not be applied automatically on the next user login. To inherit changed quota values, the option *Reapply settings on every login* can be activated at the quota policy.

Quota policies can only be used on partitions for which the quota support is enabled in the UMC module, see *Activating file system quota* (page 246).

Note: File system quotas always apply to a full partition. Even if the policies are defined for shares, they are used on complete partitions. If, for example, three shares are provided on one server which are all saved on the separate `/var/` partition and three different policies are configured and used, the most restrictive setting applies for the complete partition. If different quotas are used, it is recommended to distribute the data over individual partitions.

12.4.3 Evaluation of quota during login

The settings defined in the UCS management system are evaluated and enabled during login to UCS systems by the tool **univention-user-quota** run in the PAM stack.

If no quota are needed, the evaluation can be disabled by setting the Univention Configuration Registry Variable *quota/userdefault* (page 303) to no.

If the Univention Configuration Registry Variable *quota/logfile* (page 303) is set to any filename, the activation of the quotas is logged in the specified file.

12.4.4 Querying the quota status by administrators or users

A user can view the quota limits defined for a system using the command **repquota -va**, e.g.:

```
*** Report for user quotas on device /dev/vdb1
Block grace time: 7days; Inode grace time: 7days
                        Block limits                File limits
User            used    soft    hard  grace    used  soft  hard  grace
----------------------------------------------------------------------
root            --      20       0       0              2     0     0
Administrator   --       0       0  102400              0     0     0
user01          --  234472 2048000 4096000              2     0     0
user02          --       0 2048000 4096000              0     0     0

Statistics:
Total blocks: 8
Data blocks: 1
Entries: 4
Used average: 4.000000
```

Logged in users can use the **quota -v** command to view the applicable quota limits and the current utilization.

Further information on the commands can be found in the man pages of the commands.

CHAPTER

THIRTEEN

PRINT SERVICES

Univention Corporate Server includes a print system, which can also be used to realize complex environments. Printers and printer groups can be created and configured conveniently in the UMC module *Printers*.

The print services are based on *CUPS (Common Unix Printing System)*. CUPS manages print jobs in print queues and converts print jobs into the native formats of the connected printers. The print queues are administrated via the UMC module *Print jobs*, see *Administration of print jobs and print queues* (page 254).

All printers set up in CUPS can be directly used by UCS systems and are automatically also provided for Windows computers when Samba is used.

The technical capacities of a printer are specified in so-called PPD files. These files include for example whether a printer can print in color, whether duplex printing is possible, whether there are several paper trays, which resolutions are supported and which printer control languages are supported (e.g., PCL or PostScript).

Print jobs are transformed by CUPS with the help of filters into a format that the respective printer can interpret, for example into PostScript for a PostScript-compatible printer.

UCS already includes a wide variety of filters and PPD files. Consequently, most printers can be employed without the need to install additional drivers. The setting up of additional PPD files is described in *Integrating additional PPD files* (page 259).

A printer can either be connected directly to the print server locally (e.g., via the USB port or a parallel port) or communicate with a printer via remote protocols (e.g., TCP/IP compatible printers, which are connected via `IPP` or `LPD`).

Network printers with their own IP address should be registered in the UMC module *Computers* as an IP client (see *UCS system roles* (page 33)).

CUPS offers the possibility of defining printer groups. The included printers are used employed alternating, which allows automatic load distribution between neighboring printers.

Print shares from Windows systems can also be integrated in the CUPS print server, see *Creating a printer share* (page 250).

13.1 Installing a print server

A print server can be installed from the Univention App Center with the application **`Print server (CUPS)`**. Alternatively, the software package **`univention-printserver`** can be installed (**`univention-run-join-scripts`** must be executed after installation). Additional information can be found in *Installation of further software* (page 99).

13.2 Setting the local configuration properties of a print server

The configuration of the CUPS print server is performed via settings from the LDAP directory service and Univention Configuration Registry. If the Univention Configuration Registry Variable *cups/include/local* (page 292) is set to `true`, the `/etc/cups/cupsd.local.conf` file is included, in which arbitrary options can be defined. Changes in this file require **`ucr commit /etc/cups/cupsd.conf`** to get applied.

If an error occurs when working through a printer queue (e.g., because the connected printer is switched off), the further processing of the queue is stopped by default. This must then be reactivated by the administrator (see *Administration of print jobs and print queues* (page 254)). If the Univention Configuration Registry Variable *cups/errorpolicy* (page 292) is set to `retry-job`, CUPS automatically attempts to process unsuccessful print jobs again every 30 seconds.

13.3 Creating a printer share

Print shares are administrated in the UMC module *Printers* with the *Printer share* object type (see *Univention Management Console modules* (page 69)).

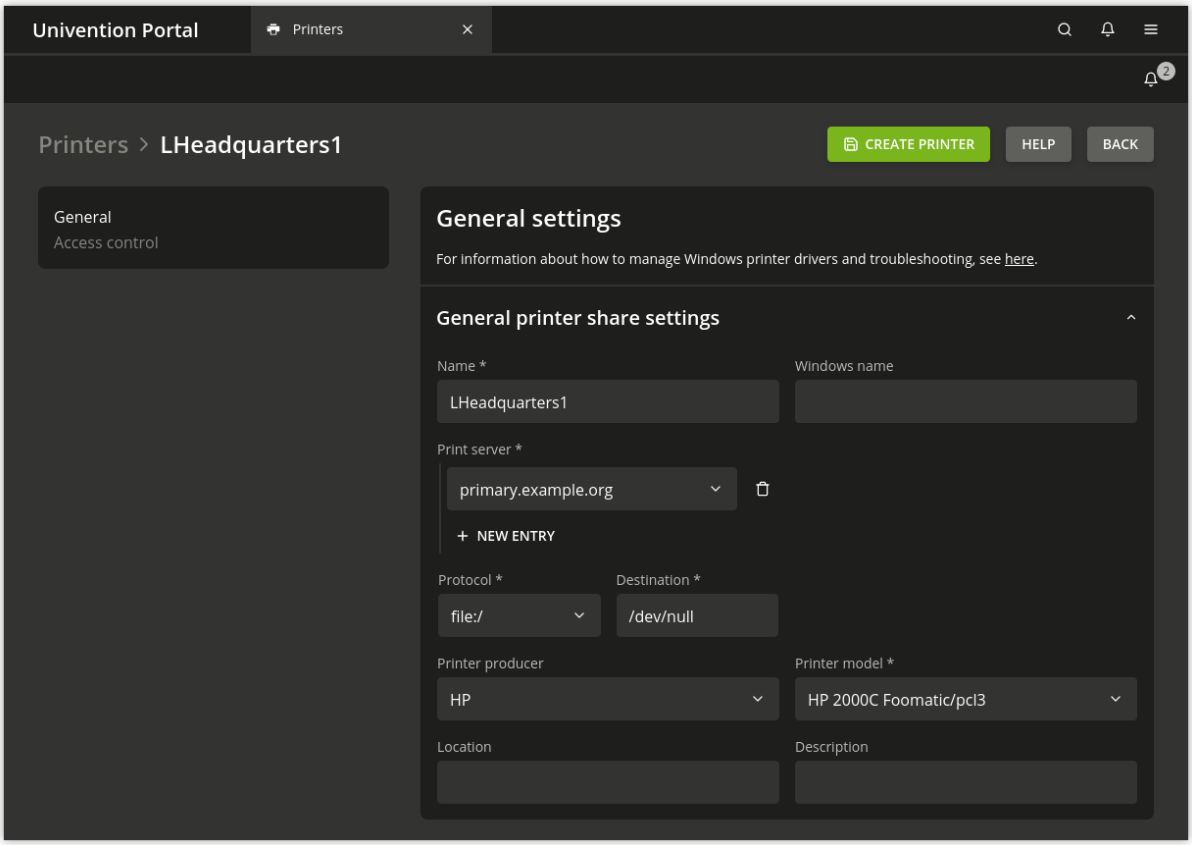


Fig. 13.1: Creating a printer share

When adding/deleting/editing a printer share, the printer is automatically configured in CUPS. CUPS does not have an LDAP interface for printer configuration, instead the `printers.conf` file is generated via a listener module. If Samba is used, the printer shares are also automatically provided for Windows clients.

13.3.1 Printers UMC module - General tab

Table 13.1: *General* tab

Attribute	Description
Name (*)	This input field contains the name of the printer share, which is used by CUPS. The printer appears under this name in Linux and Windows. The name may contain alphanumeric characters (i.e., uppercase and lowercase letters `a` to `z` and numbers `0` to `9`) as well as hyphens and underscores. Other characters (including blank spaces) are not permitted.
Print server (*)	A print server manages the printer queue for the printers to be shared. It converts the data to be printed into a compatible print format when this is necessary. If the printer is not ready, the print server saves the print jobs temporarily and forwards them on to the printer subsequently. If more than one print server is specified, the print job from the client will be sent to the first print server to become available. Only UCS Directory Node and Managed Nodes on which the **univention-printserver** package is installed are displayed in the list.
Protocol and Destination (*)	These two input fields specify how the print server accesses the printer. The following list describes the syntax of the individual protocols for the configuration of printers connected locally to the server: • `parallel://`*`devicefile`* Example: `parallel://dev/lp0` • `socket://`*`server`*`:`*`port`* Example: `socket://printer_03:9100` • `usb://`*`devicefile`* Example: `usb://dev/usb/lp0` The following list describes the syntax of the individual protocols for the configuration of network printers: • `http://`*`server`*`[:`*`port`*`]/`*`path`* Example: `http://192.0.2.10:631/printers/remote` • `ipp://`*`server`*`/printers/`*`queue`* Example: `ipp://printer_01/printers/xerox` • `lpd://`*`server`*`/`*`queue`* Example: `lpd://192.0.2.30/bwdraft` The `cups-pdf` protocol is used for integrating a pseudo printer, which creates a PDF document from all the print jobs. The setup is documented in *Generating PDF documents from print jobs* (page 255). The `file:/` protocol expects a filename as a target. The print job is then not sent to the printer, but instead written in this file, which can be useful for test purposes. The file is rewritten with every print job. The `smb://` protocol can be used to mount a Windows print share. For example, to integrate the `laser01` printer share from Windows system `win01`, `win01/laser01` must be specified as destination. The manufacturer and model must be selected according to the printer in question. The print server uses the printer model settings to convert the print jobs where necessary and send these directly to the URI `smb://win01/laser01`. No Windows drivers are used in this. Independent of these settings, the printer share can be mounted by other Windows systems with the corresponding printer drivers.
Manufacturer	When the printer manufacturer is selected, the *Printer model* selection list updates automatically.
Printer model (*)	This selection list shows all the printers PPD files available for the selected manufacturer. If the required printer model is not there, a similar model can be selected and a test print used to establish correct function. *Integrating additional PPD files* (page 259) explains how to expand the list of printer models.
Samba name	A printer can also be assigned an additional name by which it can be reached from Windows. Unlike the CUPS name (see *Name*), the Samba name may contain blank spaces and umlauts. The printer is then available to Windows under both the CUPS name and the Samba name. Using a Samba name in addition to the CUPS name is practical, for example, if the printer was already in use in Windows under a name which contains blank spaces or umlauts. The printer can then still be reached under this name without the need to reconfigure the Windows computers.
Location	This data is displayed by some applications when selecting the printer. It can be filled with any text.

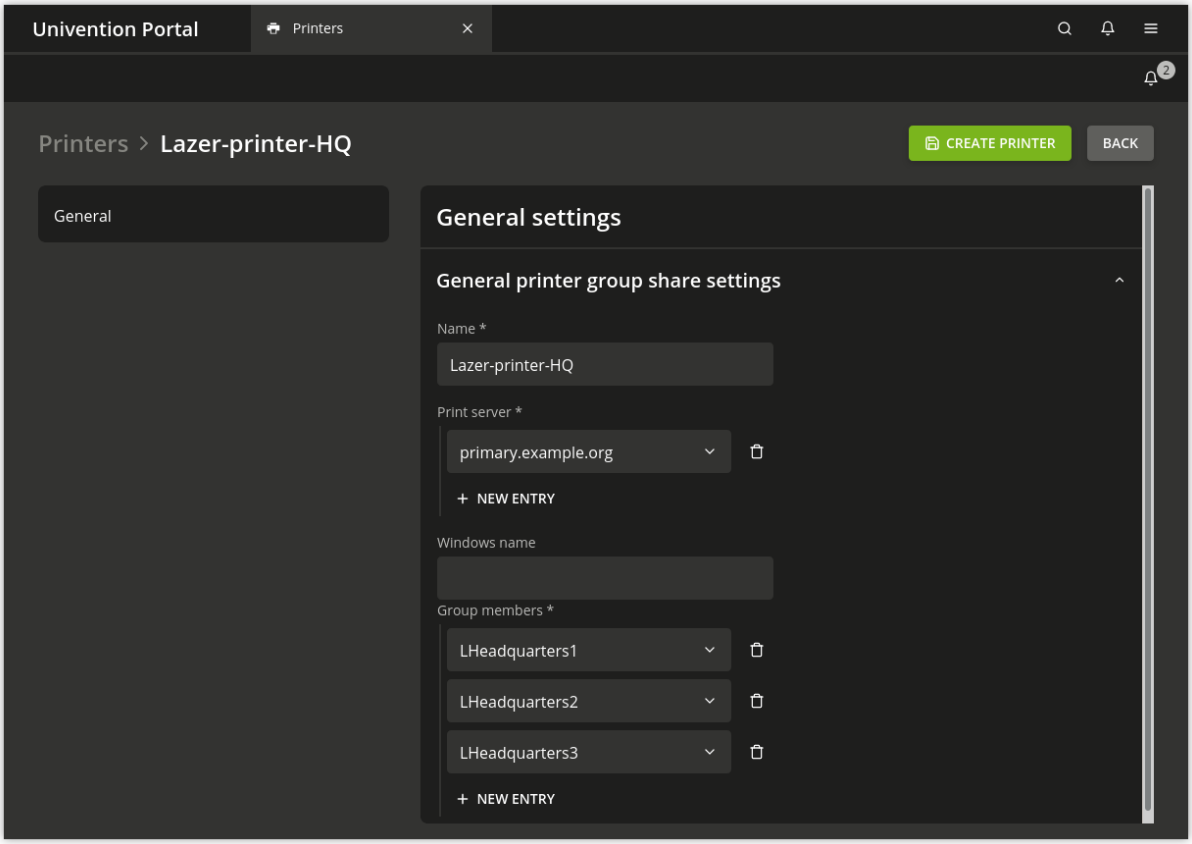
13.3.2 Printers UMC module - Access control tab

Table 13.2: *Access control* tab

Attribute	Description
Access control	Access rights for the printer can be specified here. Access can be limited to certain groups or users or generally allowed and certain groups or users blocked specifically. As standard, access is available for all groups and users. These rights are also adopted for the corresponding Samba printer shares, so that the same access rights apply when printing via Samba as when printing directly via CUPS. This access control is useful for the management of printers spread across several locations, so that the users at location A do not see the printers of location B.
Allowed/denied users	This lists individual users for whom access should be controlled.
Allowed/denied groups	This lists individual groups for whom access should be controlled.

13.4 Creating a printer group

CUPS offers the possibility to group printers into classes. These are implemented in UCS as *printer groups*. Printer groups appear to clients as normal printers. The aim of such a printer group is to create a higher availability of printer services. If the printer group is used to print, the job is sent to the first printer in the printer group to become available. The printers are selected based on the round robin principle so that the degree of utilization is kept uniform.



A printer group must have at least one printer as a member. Only printers from the same server can be members of the group.

Caution: The possibility of grouping printers shares from different printer servers in a printer group makes it possible to select printer groups as members of a printer group. This could result in a printer group adopting itself as a group member. This must not be allowed to happen.

Printer groups are administrated in the UMC module *Printers* with the *Printer share* object type (see *Univention Management Console modules* (page 69)).

Table 13.3: *General* tab

Attribute	Description
Name (*)	This input field contains the names of the printer group share, which is used by CUPS. The printer group appears under this name in Linux and Windows. The name may contain alphanumeric characters (i.e., uppercase and lowercase letters `a` to `z` and numbers `0` to `9`) as well as hyphens and underscores. Other characters (including blank spaces) are not permitted.
Print server (*)	A range of print servers (spoolers) can be specified here to expand the list of printers available for selection. Printers which are assigned to the servers specified here can then be adopted in the *Group members* list from the selection arranged below them.
Samba name	A printer group can also be assigned an additional name by which it can reached from Windows. Unlike the CUPS name (see *Name*), the Samba name may contain blank spaces and umlauts. The printer is then available to Windows under both the CUPS name and the Samba name. Using a Samba name in addition to the CUPS name is practical, for example, if the printer group was already in use in Windows under a name which contains blank spaces or umlauts. The printer group can then still be reached under this name without the need to reconfigure the Windows computers.
Group members	This list is used to assign printers to the printer group.

13.5 Administration of print jobs and print queues

The UMC module *Print jobs* allows you to check the status of the connected printers, restart paused printers and remove print jobs from the queues on printer servers.

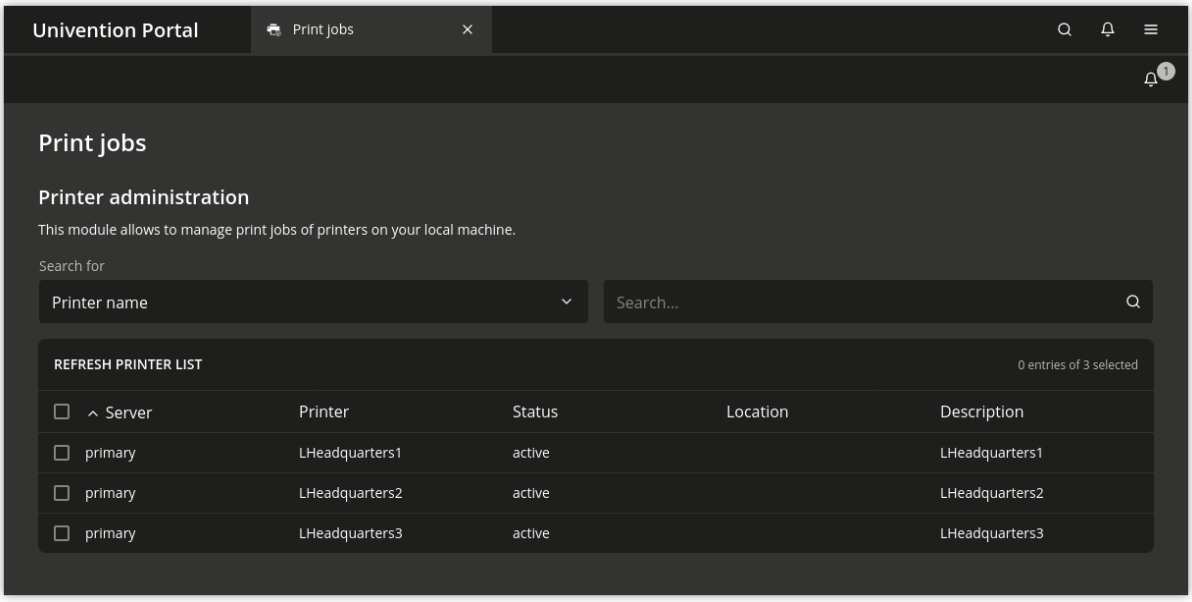


Fig. 13.2: Printer administration

The start page of the module contains a search mask with which the available printers can be selected. The result list displays the server, name, status, location and description of the respective printer. The status of more than one printer can be changed simultaneously by selecting the printers and running either the *deactivate* or *activate* function.

Clicking on the printer name displays details of the selected printer. The information displayed includes a list of the print jobs currently in the printer queue. These print jobs can be deleted from the queue by selecting the jobs and running the *Delete* function.

13.6 Generating PDF documents from print jobs

Installing the **univention-printserver-pdf** package expands the print server with a special *cups-pdf* printer type, which converts incoming print jobs into PDF documents and adds them in a specified directory on the printer server where they are readable for the respective user. After the installation of the package, **univention-run-join-scripts** must be run.

The `cups-pdf:/` protocol must be selected when creating a PDF printer in the UMC module *Printers* (see *Creating a printer share* (page 250)); the destination field remains empty.

`PDF` must be selected as *Printer producer* and `Generic CUPS-PDF Printer` as *Printer model*.

The target directory for the generated PDF documents is set using the Univention Configuration Registry Variable *cups/cups-pdf/directory* (page 292). As standard it is set to `/var/spool/cups-pdf/%U` so that **cups-pdf** uses a different directory for each user.

Print jobs coming in anonymously are printed in the directory specified by the Univention Configuration Registry Variable *cups/cups-pdf/anonymous* (page 292) (standard setting: `/var/spool/cups-pdf/`).

By default generated PDF documents are kept without any restrictions. If the Univention Configuration Registry Variable *cups/cups-pdf/cleanup/enabled* (page 292) is set to `true`, old PDF print jobs are deleted via a Cron job. The storage time in days can be configured using the Univention Configuration Registry Variable *cups/cups-pdf/cleanup/keep* (page 292).

13.7 Mounting of print shares in Windows clients

The printer shares set up in the UMC module *Printers* can be added as network printers on Windows systems. This is done via the Control Panel under *Add a device* ▸ *Add a printer*. The printer drivers need to be set up during the first access. If the drivers are stored on the server side (see below), the drivers are assigned automatically.

Printer shares are usually operated using the Windows printer drivers provided. The network printer can alternatively be set up on the Windows side with a standard PostScript printer driver. If a color printer should be accessed, a driver for a PostScript-compatible color printer should be used on the Windows side, e.g., *HP Color LaserJet 8550*.

Caution: The printer can only be accessed by regular users when they have local permissions for driver installation or the respective printer drivers were stored on the printer server. If this is not the case, Windows may issue an error warning that the permissions are insufficient to establish a connection to the printer.

Windows supports a mechanism for providing the printer drivers on the print server (*Point 'n' Print*). The following guide describes the provision of the printer drivers in Windows for a print share configured in the UMC module *Printers*. Firstly, the printer drivers must be stored on the print server. There are a number of pitfalls in the Windows user wizard, so it is important to follow the individual steps precisely.

1. Firstly, the printer drivers must be downloaded from the manufacturer's website. If you are using an environment in which 64-bit installations of Windows are used, you will need both versions of the drivers (32 and 64 bit). The `INF` files are required.
2. Now you need to start the **printmanagement.msc** program. Clicking on *Add/remove server* in the *Action* menu item allows you to add another server. The name of the printer server needs to be entered in the *Add server* input field.

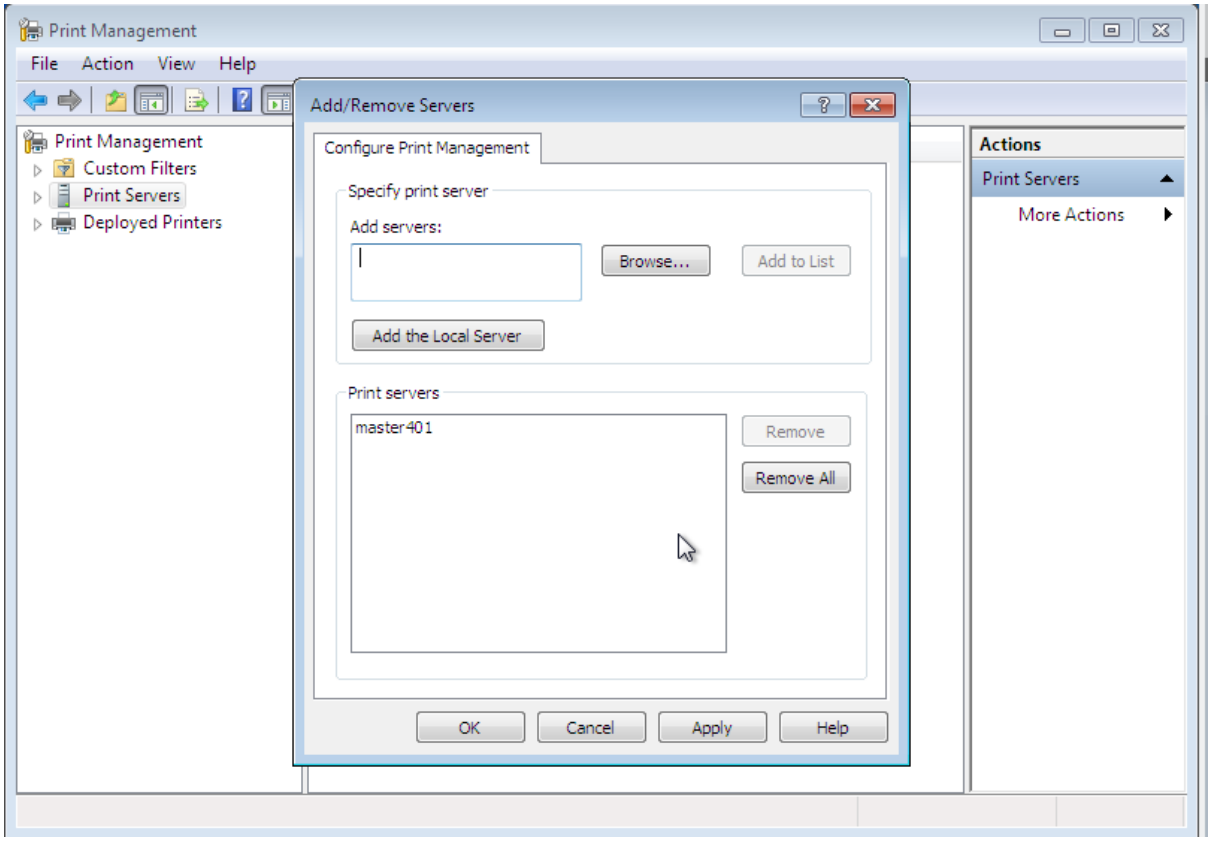


Fig. 13.3: Add printer server

3. The newly added printer server should now be listed in the print management program. Clicking on *Printers* displays the printer shares currently set up on the printer server.

4. Clicking on *Drivers* lists the saved printer drivers. Clicking on *Add driver* in the *Action* menu item opens the dialogue window for the driver installation.

   We recommend downloading the printer drivers directly from the manufacturer and selecting them during the driver installation. If you are using an environment containing 64-bit versions of Windows, start by performing a check to see if the Univention Configuration Registry Variable *samba/spoolss/architecture* (page 303) is set to `Windows x64` on the UCS Samba system. If this is not the case, both the 32-bit and the 64-bit drivers must be uploaded; if your domain only uses 64-bit Windows systems, the 32-bit driver can be ignored. The drivers for the different Windows architectures can be uploaded one after the other or together.

   If both driver architectures are selected for uploading at the same time, the 64-bit driver should be selected first in the subsequent file selection window. Once Windows has uploaded these files to the server, it asks for the location of the 32-bit drivers again. They are then also uploaded to the server.

5. After these steps the drivers are stored in the directory `/var/lib/samba/drivers/` on the print server.

6. The print share now needs to be linked to the uploaded printer driver. To do so, the list of the printers available on the printer server is opened in the **`printmanagement.msc`** program. The properties can be listed there by double-clicking on the *printer*.

7. If no printer driver is saved, a message is displayed saying that there is no printer driver installed. The prompt to install the driver should be closed with *No* here.

8. The uploaded driver now needs to be selected from the drop down menu under *Drivers* in the *Advanced* tab. Then click on *Apply* (Important: **DON'T** click on *OK*!).

9. If the printer driver in question is being assigned to a printer for the first time, a dialogue window is shown, asking whether the printer can be trusted. This should be confirmed with *Install driver*. The printer drivers saved on the server side are now downloaded to the client. If the printer driver in question has already been

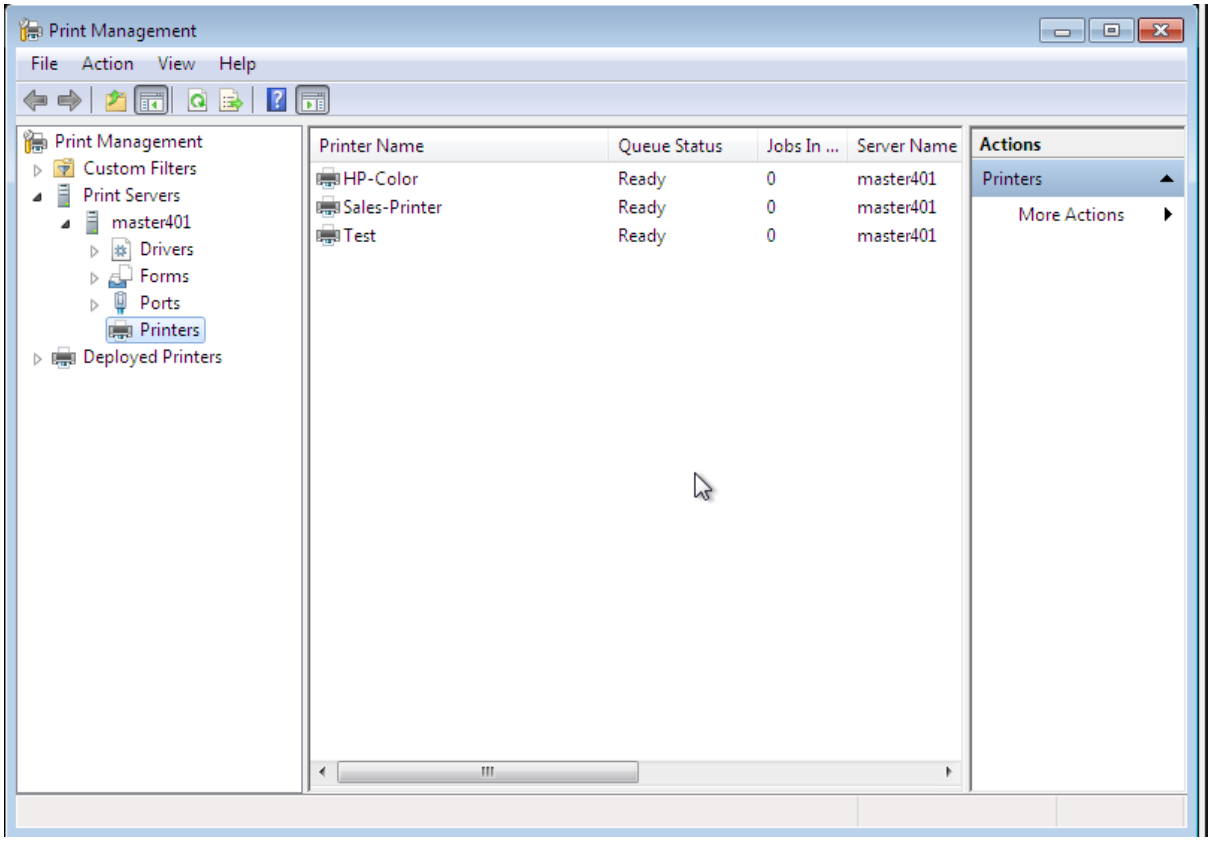


Fig. 13.4: Printer list

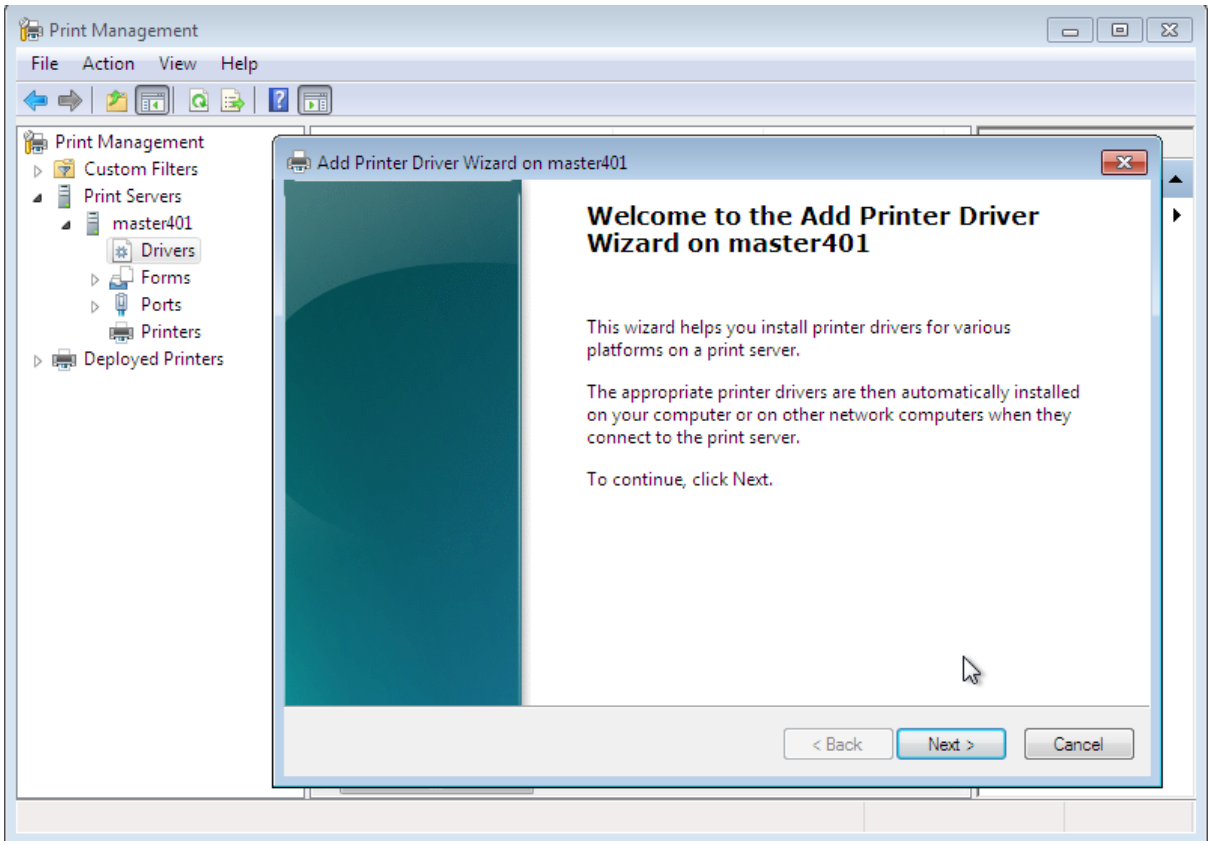


Fig. 13.5: Driver installation

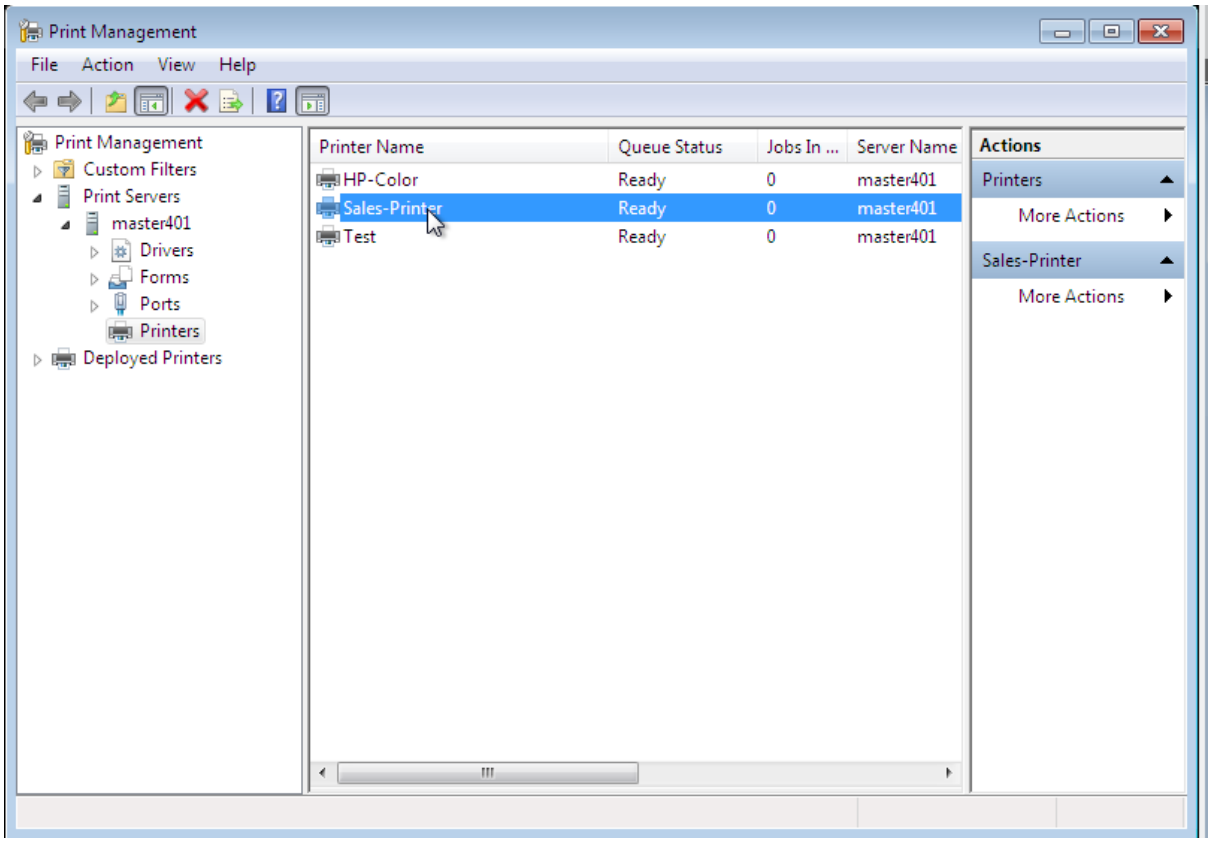


Fig. 13.6: Selecting a printer

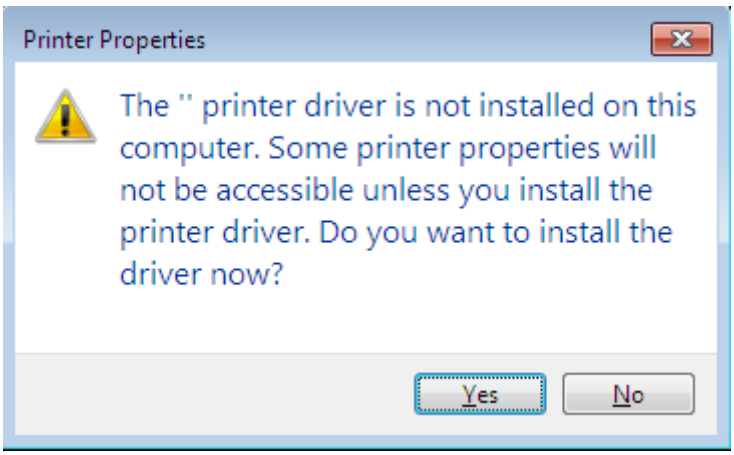


Fig. 13.7: Error message on first access

downloaded to the Windows system in question in this manner before, Windows displays an error message at this point `0x0000007a`. This can simply be ignored.

10. **Important**: Now, instead of clicking directly on *OK*, you need to return to the *General* tab again. The old name for the printer share should still be displayed on the tab.

    In UCS releases earlier than UCS 4.0-1, it is possible that the Windows system has changed the name of the printer share to the name of the printer driver. If that were accepted, the printer would no longer be associated with the share!

    If this is the case, the name of the printer on the *General* tab (the first input field next to the stylized printer symbol) needs to be reset to the name of the print share. This can be done using the *Windows name* field configured in the UMC module *Printers* (or if this was left blank, use the value from *Name*). If the name has had to be reset in this fashion, Windows then asks if you are sure that you want to change the name when *OK* is clicked. Confirm the prompt.

11. To give the Windows printer driver the opportunity to save correct standard settings for the printer, you now need to switch to the *Device settings* tab. The name of the tab differs from manufacturer to manufacturer and may also be *Settings* or even just *Configuration*.

    Clicking on *OK* closes the window. You can then print a test page. If Windows displays an error message here `0x00000006`, the printer settings must be checked again to see whether there is a manufacturer-specific tab called *Device settings* (or something similar). If so, it should be opened and then simply confirmed with *OK*. This closes the dialogue window and saves the printer drivers settings (`PrinterDriverData`) in the Samba registry.

12. At this point, it is also practical to make the settings for the paper size and other parameters, so that they are saved in the print share. Other Windows systems which subsequently access the print share will then find the correct settings automatically. These settings can usually be opened by clicking on the *Standard values…* button in the *Advanced* tab of the printer settings. The dialogue window which opens also varies from manufacturer to manufacturer. Typically, the settings for paper size and orientation are found on a tab called *Page setup* or *Paper/Quality*. Once the dialogue has been confirmed by clicking on *OK*, the printer driver saves these settings (as `Default DevMode`) for the printer in the Samba registry.

13.8 Integrating additional PPD files

The technical capabilities of a printer are specified in so-called PPD files. These files include for example whether a printer can print in color, whether duplex printing is possible, whether there are several paper trays, which resolutions are supported and which printer control languages are supported (e.g., PCL or PostScript).

In addition to the PPD files already included in the standard scope, additional ones can be added via UMC modules. The PPDs are generally provided by the printer manufacturer and need to be copied into the `/usr/share/ppd/` directory on the print servers.

The printer driver lists are administrated in the UMC module *LDAP directory*. There you need to switch to the `univention` container and then to the `cups` subcontainer. Printer driver lists already exist for the majority of printer manufacturers. These can be expanded or new ones can be added.

Table 13.4: *General* tab

Attribute	Description
Name (*)	The name of the printer driver list. The name under which the list appears in the *Printer model* selection list on the *General* tab for printer shares (see *Creating a printer share* (page 250)).
Driver	The path to the `ppd` file or to the `/usr/share/ppd/` directory. For example, if the `/usr/share/ppd/laserjet.ppd` should be used, `laserjet.ppd` must be entered here. **`gzip`** compressed files (file ending `.gz`) can also be entered here.
Description	A description of the printer driver, under which it appears in the *Printer model* selection list on the *General* tab for printer shares.

CHAPTER

FOURTEEN

MAIL SERVICES

Univention Corporate Server (UCS) provides mail services that users can access via standard mail clients such as Thunderbird.

Postfix is used for sending and receiving mails. In the basic installation, a configuration equipped for local mail delivery is set up on every UCS system. In this configuration, Postfix only accepts mails from the local server and they can also only be delivered to local system users.

The installation of the mail server component implements a complete mail transport via SMTP (see *Installation* (page 262)). Postfix is reconfigured during the installation of the component so that a validity test in the form of a search in the LDAP directory is performed for incoming emails. That means that emails are only accepted for email addresses defined in the LDAP directory or via an alias.

The IMAP service **Dovecot** is also installed on the system along with the mail server component. It provides email accounts for the domain users and offers corresponding interfaces for access via email clients. Dovecot is preconfigured for the fetching of emails via `IMAP` and `POP3`. Access via POP3 can be deactivated by setting the Univention Configuration Registry Variable *mail/dovecot/pop3* (page 298) to `no`. The same applies to IMAP and the Univention Configuration Registry Variable *mail/dovecot/imap* (page 298). The further configuration of the mail server is performed via Univention Configuration Registry, as well, see *Configuration of the mail server* (page 269).

The management of the user data of the mail server (e.g., email addresses or mailing list) is performed via UMC modules and is documented in *Management of the mail server data* (page 262). User data are stored in LDAP. The authentication is performed using a user's primary email address, i.e., it must be entered as the username in mail clients. As soon as a primary email address is assigned to a user in the LDAP directory, a listener module creates an IMAP mailbox on the mail home server. By specifying the mail home server, user email accounts can be distributed over several mail servers, as well, see *Distribution of an installation on several mail servers* (page 273).

Optionally, emails received via Postfix can be checked for spam content and viruses before further processing by Dovecot. Spam emails are detected by the classification software **SpamAssassin** (*Spam detection and filtering* (page 267)); **ClamAV** is used for the detection of viruses and other malware (*Identification of viruses and malware* (page 267)).

By default emails to external domains are delivered directly to the responsible SMTP server of that domain. Its location is performed via the resolution of the MX record in the DNS. Mail sending can also be taken over by the relay host, e.g., on the internet provider (see *Configuration of a relay host for sending the emails* (page 269)).

The UCS mail system does not offer any groupware functionality such as shared calendars or invitations to appointments. However, there are groupware systems based on UCS which integrate in the UCS management system such as Kopano and Open-Xchange. Further information can be found in the *Univention App Center* (page 92).

14.1 Installation

A mail server can be installed from the Univention App Center with the application **Mail server**. Alternatively, the software package **univention-mail-server** can be installed. Additional information can be found in *Installation of further software* (page 99). A mail server can be installed on all server system roles. The use of a UCS Directory Node is recommended because of frequent LDAP accesses.

The runtime data of the Dovecot server are stored in the `/var/spool/dovecot/` directory. If this directory is on a NFS share, please read *Mail storage on NFS* (page 274).

14.2 Management of the mail server data

14.2.1 Management of mail domains

A mail domain is an common namespace for email addresses, mailing lists and IMAP group folders. Postfix differentiates between the delivery of emails between local and external domains. Delivery to mailboxes defined in the LDAP directory is only conducted for email address from local domains. The name of a mail domain may only be composed of lowercase letters, the figures 0-9, full stops and hyphens.

Several mail domains can be managed with UCS. The managed mail domains do not need to be the DNS domains of the server - they can be selected at will. The mail domains registered on a mail server are automatically saved in the Univention Configuration Registry Variable `mail/hosteddomains` (page 299).

To ensure that external senders can also send emails to members of the domain, MX records must be created in the configuration of the authoritative name servers, which designate the UCS server as mail server for the domain. These DNS adjustments are generally performed by an internet provider.

Mail domains are managed in the UMC module *Mail* with the *Mail domain* object type.

14.2.2 Assignment of email addresses to users

A user can be assigned three different types of email addresses:

Primary email address
: The *primary email address* is used for authentication on Postfix and Dovecot. Primary email addresses must always be unique. Only one primary email address can be configured for every user. It also defines the user's IMAP mailbox. If a mail home server is assigned to a user (see *Distribution of an installation on several mail servers* (page 273)), the IMAP inbox is automatically created by a Univention Directory Listener module. The domain part of the email address must be registered in the UMC module *Mail* (see *Management of mail domains* (page 262)).

Alternative email addresses
: Emails to *alternative email addresses* are also delivered to the user's mailbox. As many addresses can be entered as you wish. The alternative email addresses do not have to be unique: if two users have the same email address, they both receive all the emails which are sent to this address. The domain part of the email address must be registered in the UMC module *Mail* (see *Management of mail domains* (page 262)). To receive emails to alternative email addresses, a user must have a primary email address.

Note: When setting the Univention Configuration Registry Variable `directory/manager/mail-address/uniqueness` to `true` the *Alternative email addresses* must be unique across the domain. No other user can have the same alternative address assigned.

Forward email addresses
: If *forward email addresses* are configured for a user, emails received through the primary or alternative email addresses are forwarded to them. A copy of the messages can optionally be stored in the user's mailbox.

Forward email addresses do not have to be unique and their domain part does not have to be registered via a UMC module.

Note: Email addresses can consist of the following characters: letters `a-z`, figures `0-9`, dots (`.`), hyphens (`-`) and underscores (_). The address has to begin with a letter and must include an `@` character. At least one mail domain must be registered for to be able to assign email addresses (see *Management of mail domains* (page 262)).

Email addresses are managed in the UMC module *Users*. The *primary email address* is entered in the *General* tab in the *User account* submenu. *Alternative email addresses* can be entered under *Advanced settings* ▸ *Mail*.

Note: Once the user account is properly configured, authentication to the mail stack is possible (`IMAP`/`POP3`/`SMTP`). Please keep in mind that after disabling the account or changing the password, the login to the mail stack is still possible for 5 minutes due to the authentication cache of the mail stack. To invalidate the authentication cache run

```
$ doveadm auth cache flush
```

on the mail server. The expiration time of the authentication cache can be configured on the mail server with the Univention Configuration Registry Variable *`mail/dovecot/auth/cache_ttl`* (page 298) and *`mail/dovecot/auth/cache_negative_ttl`* (page 298).

14.2.3 Management of mailing lists

Mailing lists are used to exchange emails in closed groups. Each mailing list has its own email address. If an email is sent to this address, it is received by all the members of the mailing list.

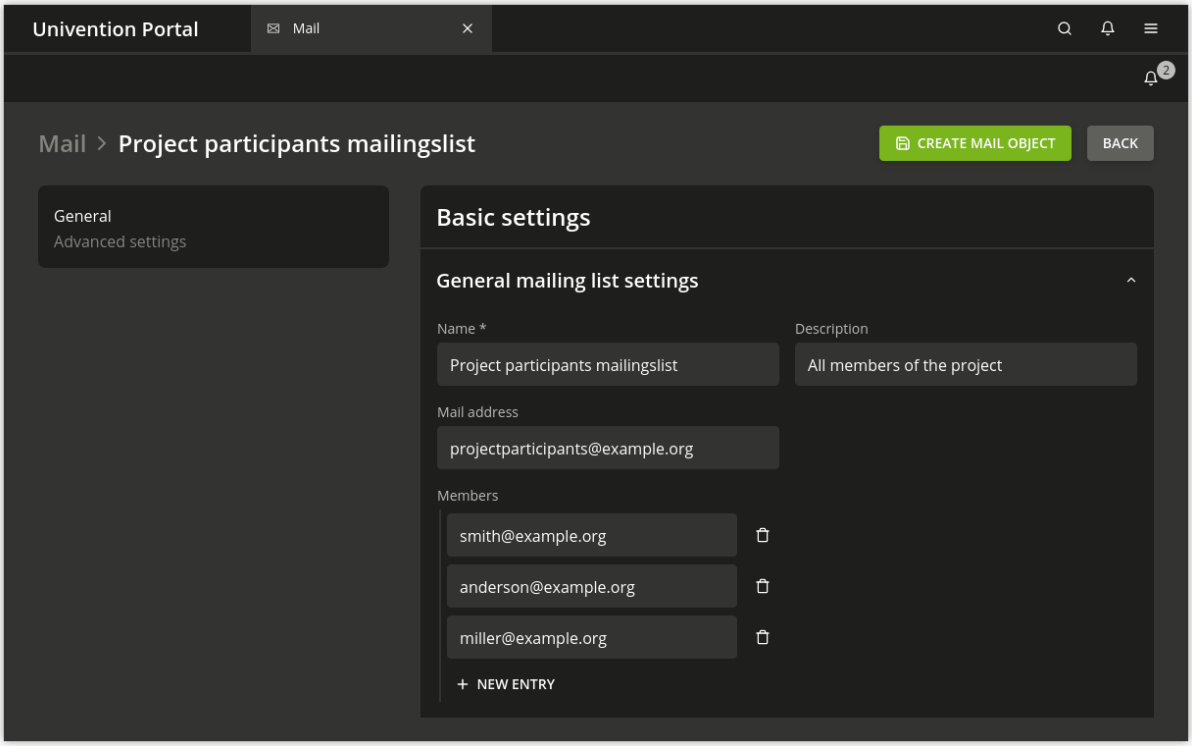


Fig. 14.1: Creating a mailing list

Mail domains are managed in the UMC module *Mail* with the *Mailing list* object type. A name of your choice can be entered for the mailing list under *Name*; the entry of a *Description* is optional. The email address of the mailing list should be entered as the *Mail address*. The domain part of the address needs to be the same as one of the

managed mail domains. As many addresses as necessary can be entered under *Members*. In contrast to mail groups (see *Management of mail groups* (page 264)), external email addresses can also be added here. The mailing list is available immediately after its creation.

By default everyone can write to the mailing list. To prevent misuse, there is the possibility of restricting the circle of people who can send mails. To do so, the Univention Configuration Registry Variable `mail/postfix/policy/listfilter` (page 299) on the mail server must be set to `yes` and Postfix restarted. *Users that are allowed to send emails to the list* and *Groups that are allowed to send emails to the list* can be specified under *Advanced settings*. If a field is set here, only authorized users/groups are allowed to send mails.

14.2.4 Management of mail groups

There is the possibility of creating a mail group: This is where an email address is assigned to a group of users. Emails to this address are delivered to the primary email address of each of the group members.

Mail groups are managed in the UMC module *Groups* (see *Group management* (page 135)).

The email address of the mail group is specified in the *mail address* input field under *Advanced settings*. The domain part of the address must be the same as one of the managed mail domains.

By default everyone can write to the mail group. To prevent misuse, there is the possibility of restricting the circle of people who can send mails. To do so, the Univention Configuration Registry Variable `mail/postfix/policy/listfilter` (page 299) on the mail server must be set to `yes` and Postfix restarted.

Users that are allowed to send emails to the group and *Groups that are allowed to send emails to the group* can be specified under *Advanced settings*. If a field is set here, only authorized users/groups are allowed to send mails.

14.2.5 Management of shared IMAP folders

Shared email access forms the basis for cooperation in many work groups. In UCS, users can create folders in their own mailboxes and assign permissions so that other users may read emails in these folders or save additional emails in them.

Alternatively, individual IMAP folders can be shared for users or user groups. This type of folder is described as a shared IMAP folder. Shared IMAP folders are managed in the UMC module *Mail* with the *Mail folder (IMAP)* object type.

Shared folders cannot be renamed, therefore the Univention Configuration Registry Variable `mail/dovecot/mailbox/rename` (page 298) is not taken into account. When a shared folder is deleted in the UMC module *Mail*, it is only deleted from the hard disk, if `mail/dovecot/mailbox/delete` (page 298) is set to `yes`. The default value is `no`.

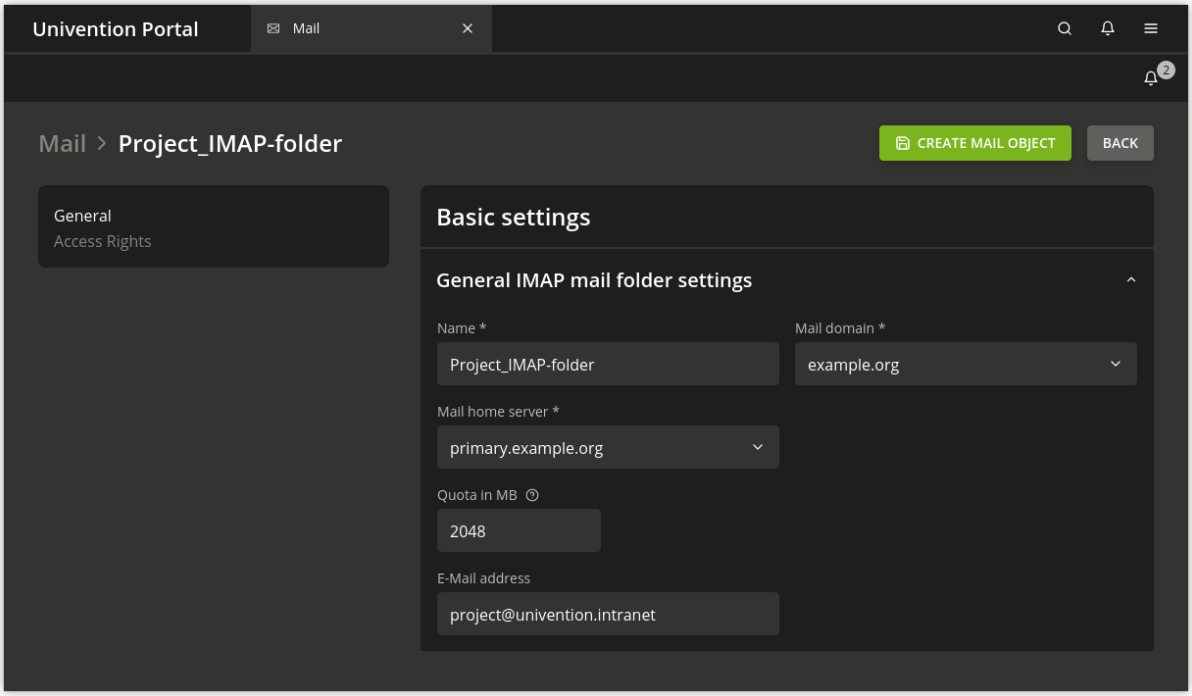


Fig. 14.2: Creating a shared IMAP folder

Shared IMAP folder - General tab

Table 14.1: *General* tab

Attribute	Description
Name (*)	The name under which the IMAP folder is available in the email clients. The name displayed in the IMAP client differs depending on if an email address is configured (see field *Email address*) or not. If no email address is configured, the IMAP folder will be displayed in the client as `name@domain/INBOX`. If an email address is configured, it will be `shared/name@domain`.
Mail domain (*)	Every shared IMAP folder is assigned to a mail domain. The management of the domains is documented in the *Management of mail domains* (page 262).
Mail home server (*)	An IMAP folder is assigned to a mail home server. Further information can be found in *Distribution of an installation on several mail servers* (page 273).
Quota in MB	This setting can be used to set the maximum total size of all emails in this folder.
Email address	An email address can be entered here via which emails can be sent directly to the IMAP folder. If no address is set here, it is only possible to write in the folder from email clients. The domain part of the email address must be registered in the UMC module *Mail* (see *Management of mail domains* (page 262)).

Shared IMAP folder - Access rights tab

Table 14.2: *Access rights* tab

Attribute	Description
Name (*)	Access permissions based on users or groups can be entered here. Users are entered with their username; the groups saved in the UMC module *Groups* can be used as groups. The access permissions have the following consequences for individual users or members of the specified group: **No access** No access is possible. The folder is not displayed in the folder list. **Read** The user may only perform read access to existing entries. **Append** Existing entries may not be edited; only new entries may be created. **Write** New entries may be created in this directory; existing entries may be edited or deleted. **Post** Sending an email to this directory as a recipient is permitted. This function is not supported by all the clients. **All** Encompasses all permissions of *write* and also allows the changing of access permissions.

14.2.6 Mail quota

The size of the users' mailboxes can be restricted via the mail quota. When this is attained, no further emails can be accepted for the mailbox by the mail server until the user deletes old mails from their account.

The limit is specified in megabytes in the *Mail quota* field under *Advanced settings* ‣ *Mail*. The default value is `0` and means that no limit is set. The multi edit mode of UMC modules can be used to assign a quota to multiple users at one time, see *Editing objects* (page 72).

The user can be warned once a specified portion of the mailbox is attained and then receives a message that their available storage space is almost full. The administrator can enter the threshold in percent and the messages subject and text:

- The threshold for when the warning message should be issued can be configured in the Univention Configuration Registry Variable *mail/dovecot/quota/warning/text* (page 299), for example `mail/dovecot/quota/warning/text/`*`PERCENT=TEXT`*

  `PERCENT` must be a number between 0 and 100 without the percent sign.

  `TEXT` will be the content of the email. If the value `TEXT` contains the string `$PERCENT`, it will be replaced in the email with the value of the limit that has been exceeded.

  The value of the Univention Configuration Registry Variable *mail/dovecot/quota/warning/subject* (page 299) will be used for the subject of the email.

- When the mail server package is installed, a subject and two warning messages are automatically configured:
  - *mail/dovecot/quota/warning/subject* (page 299) is set to `Quota-Warning`
  - `mail/dovecot/quota/warning/text/80` is set to `Your mailbox has filled up to over $PERCENT%.`

- `mail/dovecot/quota/warning/text/95` is set to `Attention: Your mailbox has already filled up to over $PERCENT%. Please delete some messages or contact the administrator.`

14.3 Spam detection and filtering

Undesirable and unsolicited emails are designated as spam. The software SpamAssassin and Postgrey are integrated in UCS for the automatic identification of these emails. SpamAssassin attempts to identify whether an email is desirable or not based on heuristics concerning its origin, form and content. Postgrey is a policy server for Postfix, which implements gray listing. Grey listing is a spam detection method which denies the first delivery attempt of external mail servers. Mail servers of spam senders most often do not perform a second delivery attempt, while legitimate servers do so. Integration occurs via the packages **univention-spamassassin** and **univention-postgrey**, which are automatically set up during the installation of the mail server package.

SpamAssassin operates a point system, which uses an increasing number of points to express a high probability of the email being spam. Points are awarded according to different criteria, for example, keywords within the email or incorrect encodings. In the standard configuration only mails with a size of up to 300 kilobytes are scanned, this can be adjusted using the Univention Configuration Registry Variable *mail/antispam/bodysizelimit* (page 297).

E-mails which are classified as spam - because they exceed a certain number of points - are not delivered to the recipient's inbox by Dovecot, but rather in the *Spam* folder below it. The name of the folder for spam can be configured with the Univention Configuration Registry Variable *mail/dovecot/folder/Spam* (page 298). The filtering is performed by a Sieve script, which is automatically generated when the user is created.

The threshold in these scripts as of which emails are declared to be spam can be configured with the Univention Configuration Registry Variable *mail/antispam/requiredhits* (page 297). The presetting (`5`) generally does not need to be adjusted. However, depending on experience in the local environment, this value can also be set lower. This will, however, result in more emails being incorrectly designated as spam. Changes to the threshold do not apply to existing users.

There is also the possibility of evaluating emails with a Bayes classifier. This compares an incoming email with statistical data already gathered from processed emails and uses this to adapt it's evaluation to the user's email. The Bayes classification is controlled by the user himself, whereby emails not identified as spam by the system can be placed in the *Spam* subfolder by the user and a selection of legitimate emails copied into the *Ham* (*mail/dovecot/folder/ham* (page 298)) subfolder. This folder is evaluated daily and data which have not yet been collected or were previously classified incorrectly are collected in a shared database. This evaluation is activated by default and can be configured with the Univention Configuration Registry Variable *mail/antispam/learndaily* (page 297).

The spam filtering can be deactivated by setting the Univention Configuration Registry Variable *mail/antivir/spam* (page 297) to `no`. When modifying Univention Configuration Registry variables concerning spam detection, the AMaViS service and Postfix must be restarted subsequently.

14.4 Identification of viruses and malware

The UCS mail services include virus and malware detection via the **univention-antivir-mail** package, which is automatically set up during the setup of the mail server package. The virus scan can be deactivated with the Univention Configuration Registry Variable *mail/antivir* (page 297).

All incoming and outgoing emails are scanned for viruses. If the scanner recognizes a virus, the email is sent to quarantine. That means that the email is stored on the server where it is not accessible to the user. The original recipient receives a message per email stating that this measure has been taken. If necessary, the administrator can restore or delete this from the `/var/lib/amavis/virusmails/` directory. Automatic deletion is not performed.

The **AMaViSd-new** software serves as an interface between the mail server and different virus scanners. The free virus scanner ClamAV is included in the package and enters operation immediately after installation. The signatures required for virus identification are procured and updated automatically and free of charge by the Freshclam service.

Alternatively or in addition, other virus scanners can also be integrated in AMaViS. Postfix and AMaViS need to be restarted following changes to the AMaViS or ClamAV configuration.

14.5 Identification of Spam sources with DNS-based Blackhole Lists

Another means of combating spam is to use a *DNS-based Blackhole List* (DNSBL) or *Real-time Blackhole List* (RBL). DNSBLs are lists of IP addresses that the operator believes to be (potential) sources of spam. The lists are checked by DNS. If the IP of the sending email server is known to the DNS server, the message is rejected. The IP address is checked quickly and in a comparatively resource-friendly manner. The check is performed *before* the message is accepted. The extensive checking of the content with SpamAssassin and anti-virus software is only performed once it has been received. Postfix has integrated support for DNSBLs[58].

DNSBLs from various projects and companies are available on the internet. Please refer to the corresponding websites for further information on conditions and prices.

The Univention Configuration Registry Variable *mail/postfix/smtpd/restrictions/recipient* (page 299) with a key-value pair *SEQUENCE=RULE* must be set to be able to use DNSBLs with Postfix: `mail/postfix/smtpd/restrictions/recipient/`*SEQUENCE=RULE*.

It can be used to configure recipient restrictions via the Postfix option `smtpd_recipient_restrictions` (see Postfix setting smtpd_recipient_restrictions[59]). The sequential number is used to sort multiple rules alphanumerically, which can be used to influences the ordering.

Tip: Existing `smtpd_recipient_restrictions` regulations can be listed as follows:

```
$ ucr search --brief mail/postfix/smtpd/restrictions/recipient
```

In an unmodified Univention Corporate Server Postfix installation, the DNSBL should be added to the end of the `smtpd_recipient_restrictions` rules. For example:

```
$ ucr set mail/postfix/smtpd/restrictions/recipient/80="reject_rbl_client ix.dnsbl.
↪manitu.net"
```

14.6 Integration of Fetchmail for retrieving mail from external mailboxes

Usually, the UCS mail service accepts mails for the users of the UCS domain directly via SMTP. UCS also offers optional integration of the software *Fetchmail* for fetching emails from external POP3 or IMAP mailboxes.

Fetchmail can be installed via the Univention App Center; simply select the **Fetchmail** application and then click on *Install*.

After the installation, the *Advanced settings* ▸ *Remote mail retrieval (single)* and *Remote mail retrieval (multi)* tabs in the user administration provide additional input fields. Use them to configure the retrieval of emails from a remote mail server.

Fetchmail delivers emails to the inboxes of the corresponding users. The user account must have the primary email address configured for this. Before using multi-drop configurations, read THE USE AND ABUSE OF MULTIDROP MAILBOXES[60] in the Fetchmail manual.

The mail is fetched every twenty minutes once at least one email address is configured for mail retrieval. After the initial configuration of a user Fetchmail needs to be started in the UMC module *System services*. In that module the

[58] http://www.postfix.org/postconf.5.html#reject_rbl_client
[59] http://www.postfix.org/postconf.5.html#smtpd_recipient_restrictions
[60] https://www.fetchmail.info/fetchmail-man.html#the-use-and-abuse-of-multidrop-mailboxes

fetching can also be disabled (alternatively by setting the Univention Configuration Registry Variable *fetchmail/autostart* (page 293) to `false`).

Table 14.3: *Remote mail retrieval (single)* tab

Attribute	Description
Username	The username to connect to the email server for fetching emails.
Password	The password for the user to connect to the email server for fetching mail.
Protocol	The protocol that Fetchmail uses for fetching emails. Choose either `IMAP` or `POP3`.
Remote mail server	The hostname of the email server that Fetchmail uses to fetch emails.
Use SSL	This option enables encrypted mail fetching. For it to work, this feature has to be supported by the mail server.
Keep mail on remote server	By default, Fetchmail deletes fetched email from the remote server after the transfer. To keep the emails on the remote server, enable this option.

Table 14.4: *Remote mail retrieval (multi)* tab

Attribute	Description
Username	The username to connect to the email server for fetching emails.
Password	The password for the user to connect to the email server for fetching mail.
Protocol	The protocol that Fetchmail uses for fetching emails. Choose either `IMAP` or `POP3`.
Remote mail server	The hostname of the email server that Fetchmail uses to fetch emails.
Local Domain Names	A space-separated list of local domain names. Leave it empty to use all local domains.
Virtual *qmail* prefix	Fetchmail removes the defined string prefix from the email address found in the header specified with the envelope header option. For example, if the value is `example-prefix-` and Fetchmail retrieves an email whose header matches an address such as `example-prefix-info@remotedomain.com`, Fetchmail forwards the email as `info@localdomain.com`.
Envelope Header	The value of this field sets the header that Fetchmail expects to appear as a copy of the mail envelope address. Fetchmail uses it for mail rerouting.
Use SSL	This option enables encrypted mail fetching. For it to work, this feature has to be supported by the mail server.
Keep mail on remote server	By default, Fetchmail deletes fetched email from the remote server after the transfer. To keep the emails on the remote server, enable this option.

14.7 Configuration of the mail server

This section describes the configuration of the **Postfix** mail server in UCS.

14.7.1 Configuration of a relay host for sending the emails

By default, **Postfix** creates a direct SMTP connection to the mail server responsible for the domain when sending mail to a non-local address.

Alternatively, **Postfix** can use a mail relay server. This is a server that receives mail and handles its transport. Administrators may have this type of mail relay server, such as one provided by the company's headquarters or internet provider.

To set up a relay host, specify it as a fully qualified domain name (FQDN) in the Univention Configuration Registry Variable *mail/relayhost* (page 300).

Examples:

- `ucr set mail/relayhost="mx01.example.com"`

  The mail server delivers outgoing mail to the mail server `mx01.example.com` on port `25`.

- `ucr set mail/relayhost="mx01.example.com:587"`

  The mail server delivers outgoing mail to the mail server `mx01.example.com` on port `587`.

Postfix determines the actual target address of the relay mail server by querying the MX/SRV record in the DNS. To turn off MX lookup, use the format *[FQDN relay host]*, as you see in the following examples:

- `ucr set mail/relayhost="[mx01.example.com]"`

  The mail server delivers outgoing mail to the mail server `mx01.example.com` on port `25`.

- `ucr set mail/relayhost="[mx01.example.com]:587"`

  The mail server delivers outgoing mail to the mail server `mx01.example.com` on port `587`.

If the relay host requires authentication for sending, set the Univention Configuration Registry Variable *mail/relayauth* (page 300) to `yes` and edit the `/etc/postfix/smtp_auth` file. Provide the FQDN for the relay host, username, and password in this file in one line in the format: *FQDN relay host username*`:`*password*. The part for the *FQDN relay host* must look exactly like the value of *mail/relayhost* (page 300).

Examples:

- `mx01.example.com:587 outgoing-username@example.com:verySecretPassword`
- `[mx01.example.com]:587 outgoing-username@example.com:verySecretPassword` with MX lookup turned off

To adopt the changes in **Postfix**, complete the following commands:

1. Update the authentication mapping:

```
$ postmap /etc/postfix/smtp_auth
```

2. If you changed *mail/relayauth* (page 300), you must update the TLS policy mapping file:

```
$ postmap /etc/postfix/tls_policy
```

3. If you changed *mail/relayhost* (page 300), you must tell the mail server to reload the configuration:

```
$ service postfix reload
```

Note: To ensure an encrypted connection while using a relay host, you must set the **Postfix** configuration option `smtp_tls_security_level` to `encrypt`.

Univention Corporate Server sets this option automatically, if the Univention Configuration Registry Variables *mail/relayhost* (page 300) and *mail/relayauth* (page 300) have the value `yes` and if *mail/postfix/tls/client/level* (page 300) doesn't have the value `none`.

See also:

***postconf(5)* - manual page for `postconf` - Postfix configuration parameters**
for a reference of the configuration values relayhost[61], and smtp_sasl_password_maps[62].

[61] https://manpages.debian.org/bookworm/postfix/postconf.5.en.html#relayhost_(default:_empty)
[62] https://manpages.debian.org/bookworm/postfix/postconf.5.en.html#smtp_sasl_password_maps_(default:_empty)

14.7.2 Configuration of the maximum mail size

The Univention Configuration Registry Variable *mail/messagesizelimit* (page 299) can be used to set the maximum size in bytes for incoming and outgoing emails. Postfix must be restarted after modifying the setting. The preset maximum size is `10240000` bytes. If the value is configured to `0` the limit is effectively removed. Please note that email attachments are enlarged by approximately a third due to the *base64* encoding.

14.7.3 Configuration of a blind carbon copy for mail archiving solutions

If the Univention Configuration Registry Variable *mail/archivefolder* (page 298) is set to an email address, Postfix sends a blind carbon copy of all incoming and outgoing emails to this address. This results in an archiving of all emails. The email address must already exist. It can be either one already registered in Univention Corporate Server as the email address of a user, or an email account with an external email service. As standard the variable is not set.

Postfix must then be restarted.

14.7.4 Configuration of soft bounces

In a number of error situations (e.g., for non-existent users) the result may be a mail bounce, i.e., the email cannot be delivered and is returned to the sender. When Univention Configuration Registry Variable *mail/postfix/softbounce* (page 299) is set to `yes` emails are never returned after a bounce, but instead are held in the queue. This setting is particularly useful during configuration work on the mail server.

14.7.5 Configuration of SMTP ports

On a Univention Corporate Server mail server Postfix is configured to listen for connections on three ports:

Port 25 - SMTP
: Port 25 (`SMTP`) should be used by other mail servers only. By default authentication is disabled. If submission of emails from users is wanted on port 25, authentication can be enabled by setting the Univention Configuration Registry Variable *mail/postfix/mastercf/options/smtp/smtpd_sasl_auth_enable* (page 299) to `yes`.

Port 465 - SMTPS
: Port 465 (`SMTPS`) allows authentication and email submission through a SSL encrypted connection. `SMTPS` has been declared deprecated in favor of port 587 but is kept enabled for legacy clients.

Port 587 - Submission
: Port 587 (`Submission`) allows authentication and email submission through a TLS encrypted connection. The use of `STARTTLS` is enforced.

The submission port should be preferred by email clients. The use of the ports `25` and `465` for email submission is deprecated.

14.7.6 Configuration of additional checks

When using a mail server that is directly accessible from the internet, there is always a risk that spam sender, spam bots or broken mail servers are continually trying to deliver unwanted emails (for example spam) to the UCS system.

To reduce the load of the mail server for such cases, Postfix brings its own service with the name **postscreen**, which is put in front of Postfix and accepts incoming SMTP connections. On these incoming SMTP connections, some lightweight tests are first performed. If the result is positive, the respective connection is passed on to Postfix. Otherwise the SMTP connection is terminated and thus the incoming mail is rejected before being in the area of responsibility of the UCS mail server.

By default, **postscreen** is not active. By setting the Univention Configuration Registry Variable *mail/postfix/postscreen/enabled* (page 299) to the value `yes`, **postscreen** can be activated.

Various UCR variables with the prefix *mail/postfix/postscreen/* (page 299) can be used to configure **postscreen**. A list of all relevant UCR variables including descriptions can be retrieved e.g. on command line via the command:

```
$ ucr search --verbose mail/postfix/postscreen/
```

Note: After each change of a UCR variable for **postscreen** the configuration of Postfix and **postscreen** should be reloaded. This can be triggered e.g. via the command **systemctl reload postfix**.

14.7.7 Custom Postfix configuration

It is possible to modify the Postfix configuration, that resides within the file `/etc/postfix/main.cf`, beyond the variables that can be set with Univention Configuration Registry Variable.

If the file `/etc/postfix/main.cf.local` exists, its content will be appended to the file `main.cf`. To transfer changes of `main.cf.local` to `main.cf`, the following command must be executed:

```
$ ucr commit /etc/postfix/main.cf
```

For the Postfix service to accept the changes, it must be reloaded:

```
$ systemctl reload postfix
```

If a Postfix variable that has previously been set in `main.cf` is set again in `main.cf.local`, Postfix will issue a warning to the log file `/var/log/mail.log`.

Note: If Postfix' behavior is not as expected, first remove configuration settings made by `main.cf.local`. Rename the file or comment out its content. Next run the two commands above. The configuration will then return to UCS defaults.

14.7.8 Configuring the alias expansion limit

When sending a mail to a group including other nested groups, the mail may not be accepted/delivered. This is caused by Postfix trying to expand the number of the primary recipients via *virtual alias expansion*. This number is limited to `1000` users by default which might be too low.

To adjust the number to (for instance) 5000 users, the following line can be added or edited in `/etc/postfix/main.cf.local`:

```
virtual_alias_expansion_limit = 5000
```

Afterwards Postfix needs to be restarted:

```
$ systemctl restart postfix
```

14.7.9 Handling of mailboxes during email changes and the deletion of user accounts

A user's mailbox is linked to the primary email address and not to the username. The Univention Configuration Registry Variable *mail/dovecot/mailbox/rename* (page 298) can be used to configure the reaction when the primary email address is changed:

- If the variable is set to `yes`, the name of the user's IMAP mailbox is changed. This is the standard setting since UCS 3.0.
- If the setting is `no`, it will not be possible to read previous emails any more once the user's primary email address is changed! If another user is assigned a previously used primary email address, they receive access to the old IMAP structure of this mailbox.

The Univention Configuration Registry Variable *mail/dovecot/mailbox/delete* (page 298) can be used to configure, whether the IMAP mailbox is also deleted. The value `yes` activates the removal of the corresponding IMAP mailbox if one of the following actions is performed:

- deletion of the user account
- removal of the primary email address from the user account
- changing the user's mail home server to a different system

With default settings (`no`) the mailboxes are kept if one of the actions above is performed.

The combination of the two variables creates four possible outcomes when the email address is changed:

Table 14.5: Renaming of email addresses

mail/dovecot/mailbox/...	Meaning
`rename=yes` and `delete=no` (default)	The existing mailbox will be renamed. Emails will be preserved and will be accessible at the new address.
`rename=yes` and `delete=yes`	The existing mailbox will be renamed. Emails will be preserved and will be accessible at the new address.
`rename=no` and `delete=no`	A new, empty mailbox will be created. The old one will be preserved on disk with the old name and will thus not be accessible to users.
`rename=no` and `delete=yes`	A new, empty mailbox will be created. The old one will be deleted from the hard disk.

14.7.10 Distribution of an installation on several mail servers

The UCS mail system offers the possibility of distributing users across several mail servers. To this end, each user is assigned a so-called mail home server on which the user's mail data are stored. When delivering an email, the responsible home server is automatically determined from the LDAP directory.

It must be observed that global IMAP folders (see *Management of shared IMAP folders* (page 264)) are assigned to a mail home server.

If the mail home server changes for a user, the user's mail data is *not* moved to the server automatically.

14.7.11 Mail storage on NFS

Dovecot supports storing emails and index files on cluster file systems and on NFS. Some settings are necessary to prevent data loss in certain situations.

The following settings assume that mailboxes are not accessed simultaneously by multiple servers. This is the case if for each user exactly one mail home server has been configured.

- *mail/dovecot/process/mmap_disable* (page 299)=yes
- *mail/dovecot/process/dotlock_use_excl* (page 299)=yes
- *mail/dovecot/process/mail_fsync* (page 299)=always

To achieve higher performance, index files can be kept on the local servers disk, instead of storing them together with the messages on NFS. The index files can then be found at /var/lib/dovecot/index/. To activate this option, set Univention Configuration Registry Variable *mail/dovecot/location/separate_index* (page 298)=yes.

With the above settings the mail server should work without problems on NFS. There are however a lot of different client and server systems in service. In case you encounter problems, here are some notes that might help:

- If NFSv2 is in use (not the case if the NFS server is a Univention Corporate Server), please set *mail/dovecot/process/dotlock_use_excl* (page 299)=no.
- If *lockd* is not in use (not the case on Univention Corporate Server systems) or if even with *lockd* in use locking error are encountered, set *mail/dovecot/process/lock_method* (page 298)=dotlock. This does lower the performance, but solves most locking related errors.
- Dovecot flushes NFS caches when needed if you set *mail/dovecot/process/mail_nfs_storage* (page 298)=yes, but unfortunately this doesn't work 100%, so you can get random errors. The same holds for flushing NFS caches after writing index files with *mail/dovecot/process/mail_nfs_index* (page 298)= yes.

See also:

Mail Location Settings[Page 274, 63] **in the Dovecot documentation**
for more information about mailbox locations.

Shared mailboxes[64] **in the Dovecot documentation**
for more information about mailbox sharing.

NFS[65] **in the Dovecot documentation**
for more information about using Dovecot with NFS.

14.7.12 Connection limits

In a default Univention Corporate Server configuration Dovecot allows 400 concurrent IMAP and POP3 connections each. That is enough to serve at least 100 concurrently logged in IMAP users, possibly a lot more.

How many IMAP connections are opened by a user depends on the clients they use:

- Web mail opens just a few short lived connections.
- Desktop clients keep multiple connections open over a long period of time.
- Mobile clients keep just a few connections open over a long period of time. But they tend to never close them, unnecessarily wasting resources.

The limits exist mainly to resist denial of service attacks that open a lot of connections and create lots of processes.

To list the open connections, run:

[63] https://doc.dovecot.org/2.3/configuration_manual/mail_location/
[64] https://doc.dovecot.org/2.3/configuration_manual/shared_mailboxes/
[65] https://doc.dovecot.org/2.3/configuration_manual/nfs/

```
$ doveadm who
```

To display the total amount of open connections, run:

```
$ doveadm who -1 | wc -l
```

The Univention Configuration Registry Variables *mail/dovecot/limits* (page 298)/* can be set to modify the limits. The process of adapting those variables is only semi automatic, because of their complex interaction. For the meaning of each variable refer to Dovecot documentation: Service configuration[66].

Dovecot uses separate processes for login and to access emails. The limits for these can be configured separately. The maximum number of concurrent connections to a service and the maximum number of processes for a service is also configured separately. Setting `mail/dovecot/limits/default_client_limit=3000` changes the limit for the maximum number of concurrent connections to the IMAP and POP3 services but does not change the maximum number of processes allowed to run. With the Univention Corporate Server default settings Dovecot runs in "High-security mode": each connection is served by a separate process. The default is to allow only `400` processes, so only 400 connections can be made.

To allow 3000 clients to connect to their emails, another Univention Configuration Registry Variable has to be set:

```
$ ucr set mail/dovecot/limits/default_client_limit=3000
$ ucr set mail/dovecot/limits/default_process_limit=3000
$ doveadm reload
```

Reading `/var/log/dovecot.info` reveals a warning:

```
config: Warning: service auth { client_limit=2000 } is lower than required under␣
↪max. load (15000)
config: Warning: service anvil { client_limit=1603 } is lower than required under␣
↪max. load (12003)
```

The services `auth` (responsible for login and SSL connections) and `anvil` (responsible for statistics collection) are set to their default limits. Although 3000 POP3 and IMAP connections and processes are allowed, the connection limit for the login service is too low. Leaving it like this will lead to failed logins.

The values are so high, because `default_client_limit` and `default_process_limit` do not only lift limits for IMAP and POP3, but also for other services like `lmtp` and `managesieve-login`. Those services can now start more processes that have to be monitored and can theoretically make more authentication requests. This increases the number of possible concurrent connections to the `auth` and `anvil` services.

The values have to be adapted, using the numbers from the log file:

```
$ ucr set mail/dovecot/limits/auth/client_limit=15000
$ ucr set mail/dovecot/limits/anvil/client_limit=12003
$ doveadm reload
```

Another warning appears in `/var/log/dovecot.info`:

```
master: Warning: fd limit (ulimit -n) is lower than required under max. load (2000
↪< 15000),…
 because of service auth { client_limit }
```

The Linux kernel controlled setting `ulimit` setting (limit on the number of files/connections a process is allowed to open) is changed only when the Dovecot service is restarted:

```
$ systemctl restart dovecot
```

No more warnings are written to the log file and both IMAP and POP3 servers now accept 3000 client connections each.

[66] https://doc.dovecot.org/2.3/configuration_manual/service_configuration/

Univention Corporate Server configures Dovecot to run in “High-security mode” by default. For installations with 10.000s of users, Dovecot offers the “High-performance mode”. The performance guide has further details on how to configure it, see *UCS performance guide* [5].

14.8 Configuration of mail clients for the mail server

The use of IMAP is recommended for using a mail client with the UCS mail server. STARTTLS is used to switch to a TLS-secured connection after an initial negotiation phase when using SMTP (for sending mail) and IMAP (for receiving/synchronizing mail). *Password (plain text)* should be used in combination with *STARTTLS* as the authentication method. The method may have a different name depending on the mail client. The following screenshot shows the setup of Mozilla Thunderbird as an example.

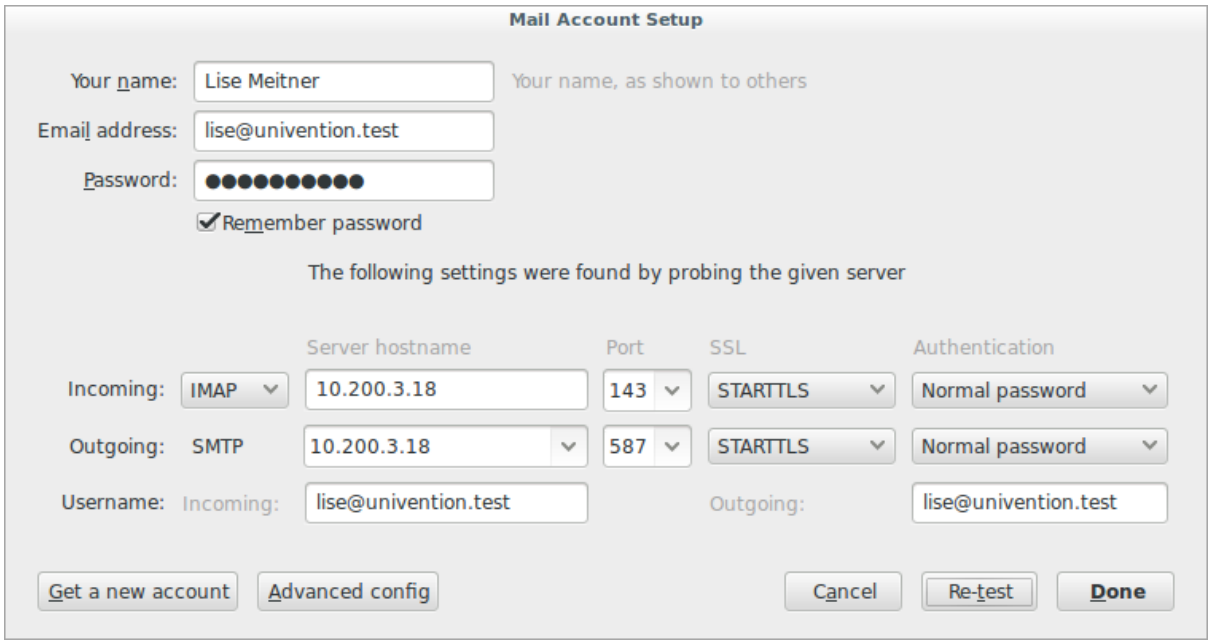


Fig. 14.3: Setup of Mozilla Thunderbird

14.9 OX Connector

OX Connector is an app in Univention App Center. It synchronizes selected users and groups to **OX App Suite** and remote installations like for example a hosted OX App Suite. Starting with **OX Connector** version `2.1.2` and **OX App Suite** version `7.10.6-ucs4`, the **OX Connector** integrates with **OX App Suite** from Univention App Center to provision user and group accounts to **OX App Suite**.

Warning: **OX App Suite** versions older than `7.10.6-ucs4` include their own synchronization. **OX Connector** doesn't synchronize with those versions and you must therefore not use it with the separate **OX App Suite** app from the App Center.

See also:

OX Connector App documentation

For more information about the **OX Connector**, refer to Integration of OX Connector and OX App Suite app[67] in the dedicated documentation at *Univention OX Connector app documentation* [16].

[67] https://docs.software-univention.de/ox-connector-app/latest/limitations.html#limit-ox-app-suite-app

CHAPTER

FIFTEEN

INFRASTRUCTURE MONITORING

UCS offers two different solutions for infrastructure monitoring.

On the one hand the UCS Dashboard helps administrators to quickly read the state of domains and individual servers. On the other hand, under UCS 4.4, with Nagios it is possible to continuously check computers and services in the background and proactively trigger a notification if a warning level is reached. From UCS 5.0-2 onward, Prometheus and Prometheus Alertmanager are used for monitoring. With UCS 5.0 support for the Nagios server component has been discontinued.

15.1 UCS Dashboard

The **`UCS Dashboard`** app allows administrators to view the state of the domain and individual servers can be read quickly and clearly on so-called dashboards. The dashboards are accessible via a web browser, access a database in the background, and deliver continuously updated reports on specific aspects of the domain or server.

15.1.1 Installation

The UCS Dashboard consists of the following parts:

UCS Dashboard
: The *UCS Dashboard* app for the visualization of data from the central Database. This component is based on the software Grafana[68][1].

UCS Dashboard Database
: The *UCS Dashboard Database* app, a time series database for storing of the metrics. This database is based on the software Prometheus.

UCS Dashboard Client
: The *UCS Dashboard Client* app for deploying the metrics of server systems. This is based on the Prometheus Node Exporter.

The app *UCS Dashboard* can be installed from the Univention App Center on a server in the domain. The installation is only supported on the system roles Primary Directory Node, Backup Directory Node or Replica Directory Node. The apps *UCS Dashboard Database* and *UCS Dashboard Client* are automatically installed on the same system.

[68] https://grafana.com/

[1] The Grafana Labs Marks are trademarks of Grafana Labs, and are used with Grafana Labs' permission. We are not affiliated with, endorsed or sponsored by Grafana Labs or its affiliates.

15.1.2 Usage

After the installation, the UCS Dashboard is linked in the portal. Alternatively, it can be accessed directly via `https://SERVERNAME-OR-IP/ucs-dashboard/`.

By default access is only granted to users of the group `Domain Admins` (e.g. the user Administrator).

Domain dashboard

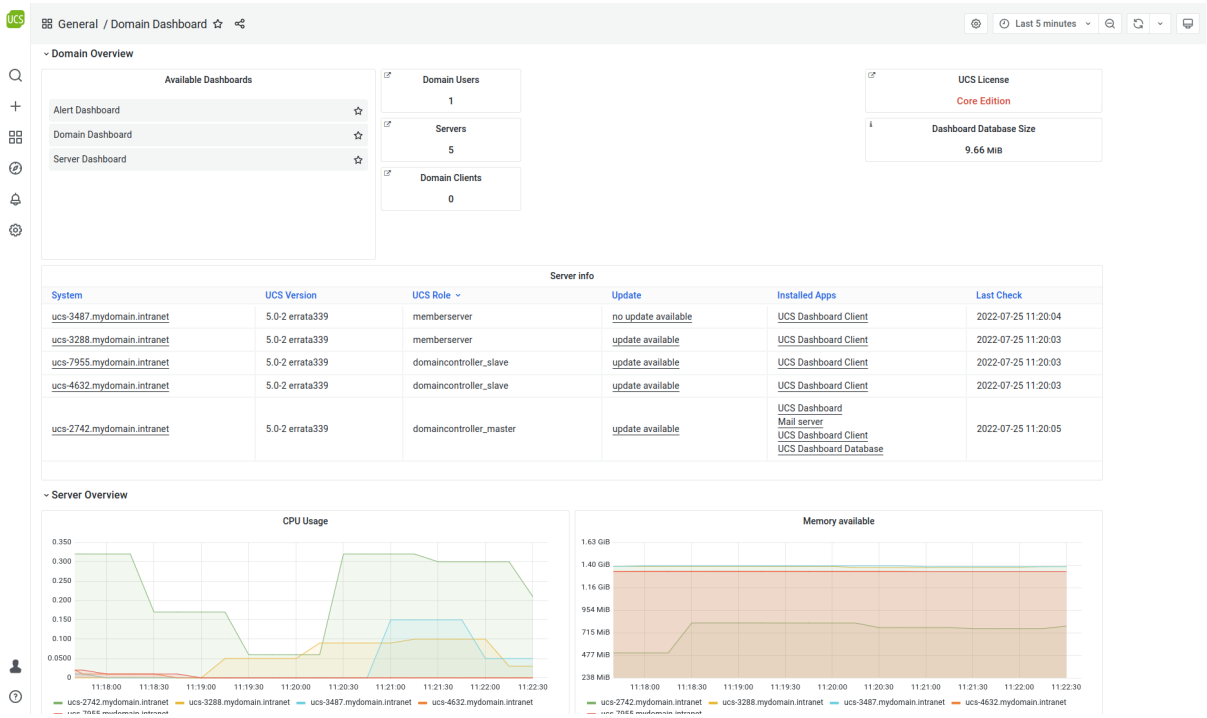


Fig. 15.1: Domain dashboard

After the login, the *Domain Dashboard* is opened by default. On this dashboard, general information about the domain is displayed, such as how many servers and how many users exist in the environment.

Furthermore, the UCS systems are listed on the dashboard, in a tabular overview, including further information, such as the server role, the installed apps or whether an update is available or not.

In addition, the CPU usage, memory usage, free hard disk space and the status of the LDAP replication are displayed. In this graphics all servers are displayed together.

Server dashboard

By default, the *Server Dashboard* is also configured. On this dashboard, detailed information about individual server systems are shown, such as the CPU- or memory usage or network throughput.

The servers can be selected in the drop down *server*. Then the graphics show the details about the selected server.



Fig. 15.2: Server dashboard

Alert dashboard

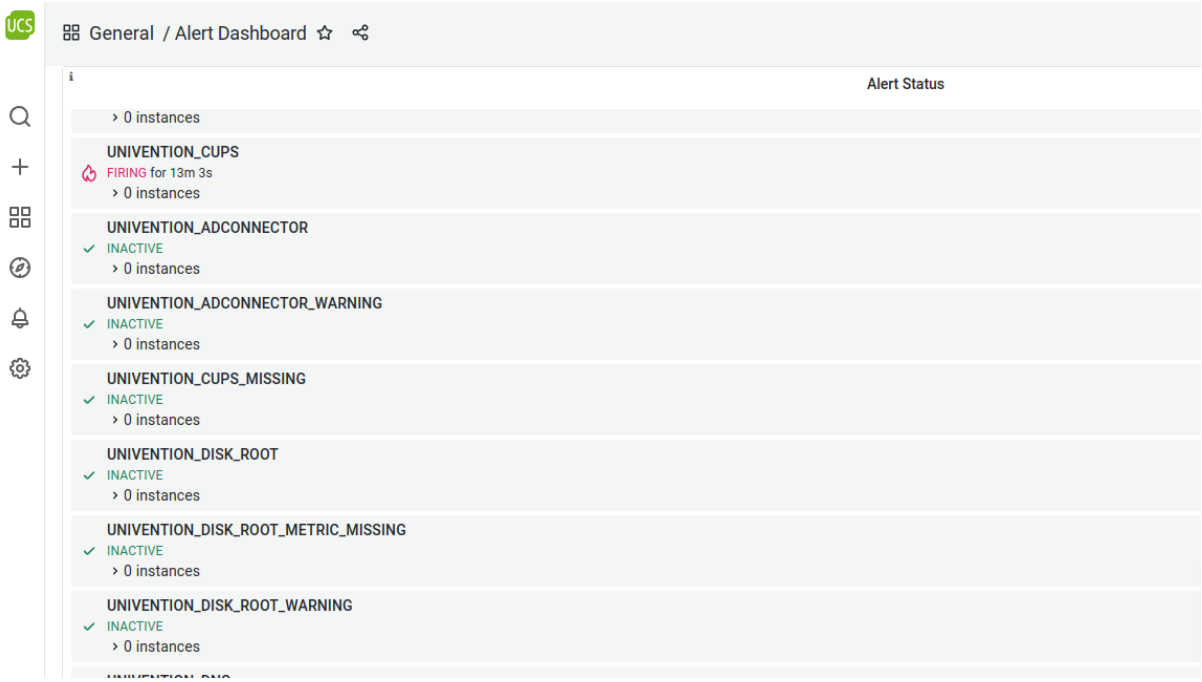


Fig. 15.3: *Alert Dashboard*

By default, the **`UCS Dashboard`** app configures the *Alert Dashboard*. The *Alert Dashboard* shows detailed information about the status of all alerts, configured in UCS.

Own dashboards

Administrators can't change the three included dashboards *Domain Dashboard*, *Server Dashboard* and *Alert Dashboard*, because Univention provides updates for them.

Instead, you can create your own dashboards. On these dashboards you can then either add already existing elements or new elements can be created. All you need to do is click on the plus sign on the left side. A new dashboard will be created which can be filled with elements.

15.2 Monitoring

New in version 5.0-2: UCS 5.0-2 supports monitoring alerts through *Prometheus* metrics.

With *Prometheus*, *Prometheus Node Exporter*, and *Prometheus Alertmanager*, administrators can verify the correct function of complex IT structures from networks, computers and services continually and automatically.

Prometheus Node Exporter exports a comprehensive collection of metrics into the Prometheus database. Besides polling system indicators like CPU, memory usage, and free disk space, they test availability and operation of different services like SSH, SMTP, and HTTP. Operation tests generally perform program steps such as the delivery of a test email or the resolution of a DNS record. The Prometheus Node Exporter provides UCS specific alerts in addition to the start metrics already included, for example an alert for the listener/notifier replication.

When the operating status changes, the monitoring informs a contact person specified in advance of the possible malfunction. In addition to the reactive notification in case of error, administrators can check the current status at any time continually in the *Grafana UCS Dashboard* web interface displaying the status information in a compact manner.

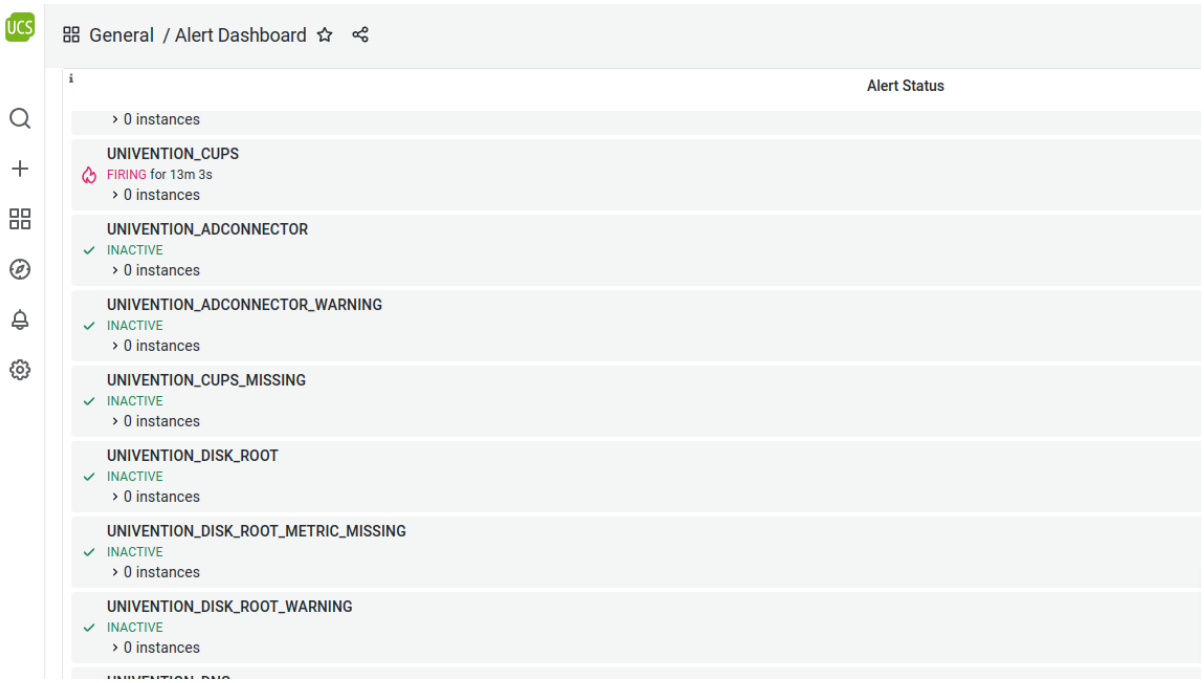


Fig. 15.4: Alert dashboard

See UCS-Dashboard *Installation* (page 277) for an overview of all involved components.

Administrators define the alert configuration in Univention Management Console. A listener module automatically generates the configuration files from information stored in the LDAP directory.

15.2.1 Installation

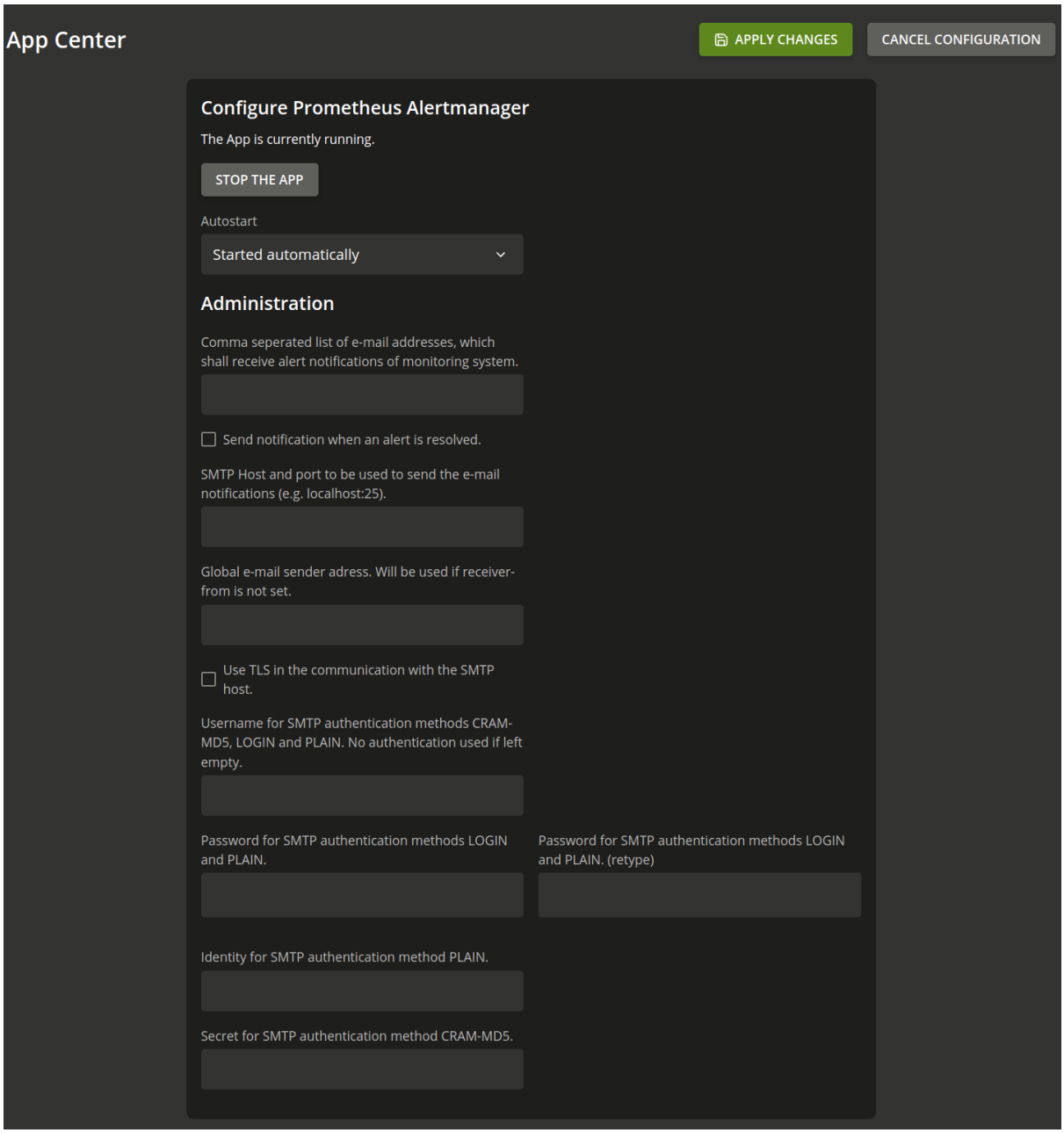
For installation of the UCS Dashboard components, see *Installation* (page 277).

Additionally to the components of the UCS Dashboard you need to install the *Prometheus Alertmanager* app and the *univention-monitoring-client*.

For every UCS system that the administrator wants to show system data on the dashboard, they must install the *UCS Dashboard Client* app. The package **univention-monitoring-client** depends on the *UCS Dashboard Client* app and is installed on every UCS system by default for the alert functionality.

Prometheus Alertmanager

The *Prometheus Alertmanager* app to send notifications for example through email for firing alerts. The Alertmanager needs some configuration to work properly.



The settings include the recipients of the email alert notifications. Furthermore, the app settings need a value for a SMTP server to send email notifications. The Alertmanager supports the SMTP authentication methods `PLAIN`, `LOGIN`, and `CRAM-MD5` as well as communication with TLS. No authentication will be used, if you leave all authentication related fields of the app settings empty.

univention-monitoring-client
: The package **univention-monitoring-client** provides standard alert plugins for checking the system health.

Administrators can install plugins with the following packages, that add alerts beyond the standard plugins provided with the **univention-monitoring-client** package:

- **univention-monitoring-raid**: Monitoring of the software RAID status
- **univention-monitoring-smart**: Test of the S.M.A.R.T. status of hard drives
- **univention-monitoring-opsi**: Test of software distribution OPSI
- **univention-monitoring-cups**: Test of CUPS printing system
- **univention-monitoring-squid**: Test of Squid proxy server
- **univention-monitoring-samba**: Test of the Samba 4 services
- **univention-monitoring-s4-connector**: Test of the S4 Connector
- **univention-monitoring-ad-connector**: Test of the AD Connector

Some services already automatically setup their respective package during installation. For example, if administrators setup the **UCS AD Connector**, it automatically includes the monitoring plugin.

15.2.2 Preconfigured monitoring checks

The installation automatically sets up basic monitoring tests for UCS systems. All alerts have label *severity* with value `critical` or `warning`.

Table 15.1: Preconfigured alerts

Alert	Description
UNIVENTION_DISK_ROOT and UNIVENTION_DISK_ROOT_WARNING	Monitors how full the / partition is. An error status is raised if the remaining free space falls below 25% or 10% by default.
UNIVENTION_DNS	Tests the function of the local DNS server and the accessibility of the public DNS server by querying the hostname www.univention.de. If no DNS forwarder is defined for the UCS domain, this request fails. In this case, www.univention.de can be replaced with the FQDN of the Primary Directory Node for example, in the monitoring/dns/lookup-domain to test the function of the name resolution.
UNIVENTION_LDAP_AUTH	Monitors the LDAP server running on UCS Directory Nodes.
UNIVENTION_LOAD and UNIVENTION_LOAD_WARNING	Monitors the system load.
UNIVENTION_NTP and UNIVENTION_NTP_WARNING	Requests the time from the NTP service on the monitored UCS system. If this deviates by more than 60 or 120 seconds, the error status is attained.
UNIVENTION_SMTP	Tests if the SMTP server is reachable. The alert fires if it is not reachable.
UNIVENTION_SSL and UNIVENTION_SSL_WARNING	Tests the remaining validity period of the UCS SSL certificates. This plugin is only suitable for Primary Directory Node and Backup Directory Node systems.
UNIVENTION_SWAP and UNIVENTION_SWAP_WARNING	Monitors the utilization of the swap partition. An error status is raised if the remaining free space falls below the threshold (40% or 20% by default).
UNIVENTION_REPLICATION and UNIVENTION_REPLICATION_WARNING	Monitors the status of the LDAP replication and recognizes the creation of a failed.ldif file and the standstill of the replication and warns of large differences between the transaction IDs.
UNIVENTION_NSCD and UNIVENTION_NSCD2	Tests the availability of the name server cache daemon (NSCD). If there is no NSCD process running, a *critical* alert is fired; if more than one process is running, a *warning* alert is fired.
UNIVENTION_WINBIND	Tests the availability of the Winbind service. If no process is running, a *critical* alert is fired.
UNIVENTION_SMBD	Tests the availability of the Samba service. If no process is running, an alert is fired.
UNIVENTION_NMBD	Tests the availability of the NMBD service, which is responsible for the NetBIOS service in Samba. If no process is running, an alert is fired.
UNIVENTION_JOINSTATUS and UNIVENTION_JOINSTATUS_WARNING	Tests the join status of a system. If a system has yet to join, a *critical* alert is fired; if non-run join scripts are available, a *warning* alert is fired.
UNIVENTION_KPASSWDD	Tests the availability of the Kerberos password service (only available on Primary/Backup Directory Nodes). If fewer or more than one process is running, an alert is fired.
UNIVENTION_PACKAGE_STATUS	Monitors the status of installed debian packages. If any package has status *half-installed* an alert is fired.
UNIVENTION_SLAPD_MDB_MAXSIZE and UNIVENTION_SLAPD_MDB_MAXSIZE_WARNING	Monitors the share of free memory pages of the *mdb* backend of SLAPD for multiple directories.
UNIVENTION_LISTENER_MDB_MAXSIZE and UNIVENTION_LISTENER_MDB_MAXSIZE_WARNING	Monitors the share of free memory pages of the *mdb* backend of SLAPD for multiple directories regarding the Univention listener.

The following monitoring alerts are only available once additional packages have been installed (see *Monitoring installation* (page 281)).

Table 15.2: Additional alerts

Alert	Description
UNIVENTION_OPSI	Monitors the OPSI daemon. If no OPSI process is running or the OPSI proxy is not accessible, the alert is fired.
UNIVENTION_SMART_SDA	Tests the S.M.A.R.T. status of the hard drive /dev/sda. Corresponding alerts exist for the hard drives sdb, sdc and sdd.
UNIVENTION_RAID and UNIVENTION_RAID_WARNING	Tests the status of the software RAID through /proc/mdadm and fires a *critical* alert if one of the hard drives in the RAID association has failed or a *warning* alert if a recovery procedure is in progress.
UNIVENTION_ADCONNECTOR and UNIVENTION_ADCONNECTOR_WARNING	Checks the status of the AD connector: • If no connector process is running, the alert is fired. • If more than one process is running per connector instance, a *warning* is fired. • If rejects occur, a *warning* alert is fired. • If the AD server can't be reached, an alert is fired. The plugin can also be used in multi-connector instances.
UNIVENTION_CUPS	Monitors the CUPS daemon. If there is no **cupsd** process running or the web interface is not accessible, a *critical* alert is fired.
UNIVENTION_SQUID	Monitors the Squid proxy. If no squid process is running or the Squid proxy is not accessible, the alert is fired.
UNIVENTION_RAID and UNIVENTION_RAID_WARNING	Monitors the status of present raid devices. The *warning* alert is fired in case of the following RAID statuses: • Rebuilding • Reconstruct • Replaced Drive • Expanding • Warning • Verify The *critical* alert is fired in case of the following RAID statuses: • Degraded • Dead • Failed • Error • Missing
UNIVENTION_S4CONNECTOR and UNIVENTION_S4CONNECTOR_WARNING	Monitors the status of Samba 4 server. A *warning* alert is fired if the Samba 4 is reachable and if any rejects are present. A *critical* alert is fired, if the server is not reachable.
UNIVENTION_SAMBA_REPLICATION	Monitors the status of the samba replication. the alert is fired if any replication failures are present.

15.2.3 Configuration

Univention Management Console offers the following settings:

- Administrators must configure the alert (see *Monitoring installation* (page 281)) and define on which computers of the domain an alert shall be active (see *Assign monitoring alerts to computers* (page 286)).
- To configure the contact person that the *Alertmanager* notifies in case of errors or alerts, set the appropriate app setting in the **Prometheus Alertmanager** app (see *Monitoring installation* (page 281)).
- Administrators can silence firing alerts for a defined time. See the Prometheus Alertmanager documentation[69]. Use the *Prometheus Alertmanager* web interface for those settings.

The basic settings already define a large number of tests for each computer, for example an alert basic configuration without the need for any further adjustments.

[69] https://prometheus.io/docs/alerting/latest/alertmanager/#silences

Configure monitoring alerts

An alert defines the monitoring of a service or a status, for example free disk space. Administrators can assign any number of computers to such an alert object.

Administrators manage monitoring alerts in the UMC module *Monitoring* with the object type *Alert*, see *Computer management module - Alerts tab* (page 148). Prometheus has no LDAP interface for the monitoring configuration. Instead, a listener module generates the configuration files when administrators add, edit, or remove alerts.

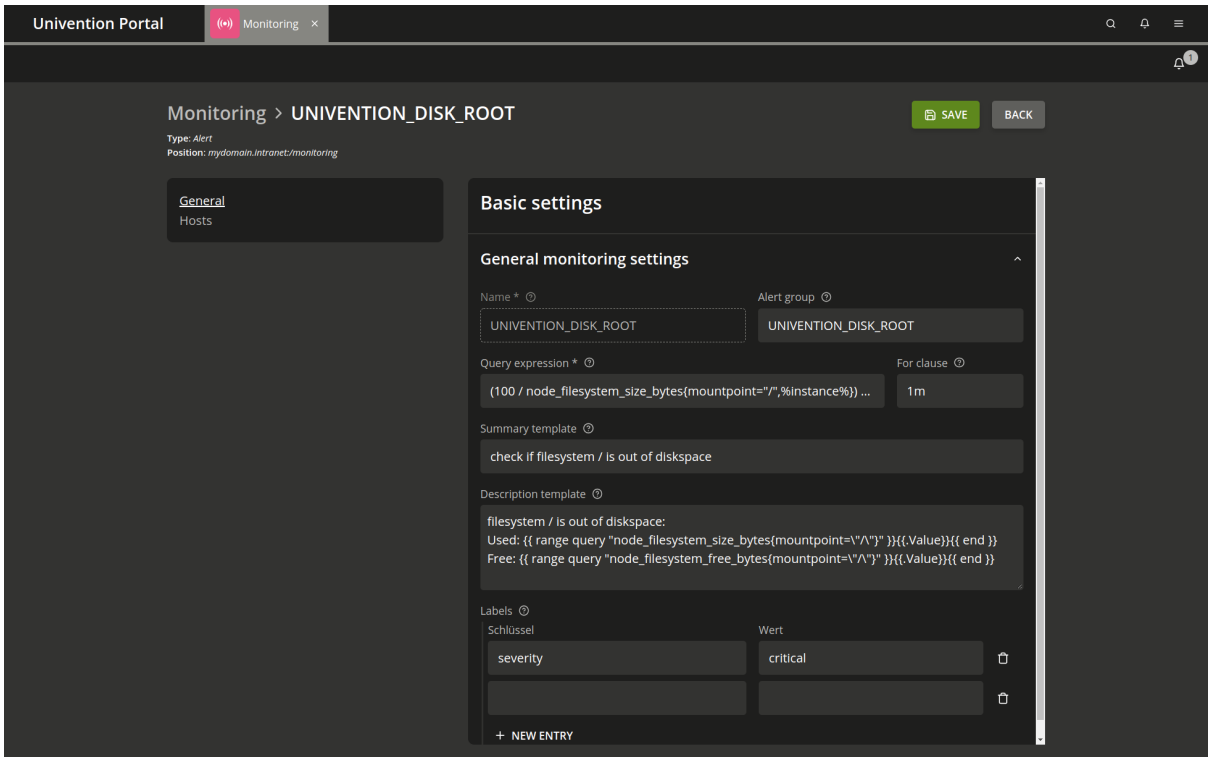


Fig. 15.5: Configuring an alert

Table 15.3: *General* tab

Attribute	Description
`Name`	An unambiguous name for the alert.
`Alert group`	Defines the group that includes the alert. Multiple alarms can belong to the same group.
`Query expression`	Prometheus query expression, which triggers the alert. The alert triggers when the given query returns a non-empty vector. For details about the syntax, see the Prometheus documentation[70].
`For clause`	Defines the time that the query expression result is non-empty until the alert triggers.
`Summary template`	The title of the alert, shown in alert dashboard and alert email notifications.
`Description template`	The description of the alert, shown in alert dashboard and alert email notifications.
`Labels`	*Prometheus* attaches labels to alerts. Labels help in queries for alerts. For example: *severity* with the value `critical` or `warning`.
`Template Values`	Query expressions, descriptions and summaries can use variable values. For example: Reference `max` through `%max%`.

[70] https://prometheus.io/docs/prometheus/latest/querying/basics/

Table 15.4: *Hosts* tab

Attribute	Description
`Assigned hosts`	*Prometheus* executes the query on the computers referenced here. The listener module runs the tests for the alert. It replaces the term `%instance%` in the query expression with a regular expression that matches the assigned hosts.

Assign monitoring alerts to computers

Prometheus can monitor all computers administered with Univention Management Console.

Navigate in the Univention Management Console to *Computers* and choose the computer you want to activate alerts on. Choose and add all alerts you like in the tab *Advanced settings* under *Alerts* and save your changes.

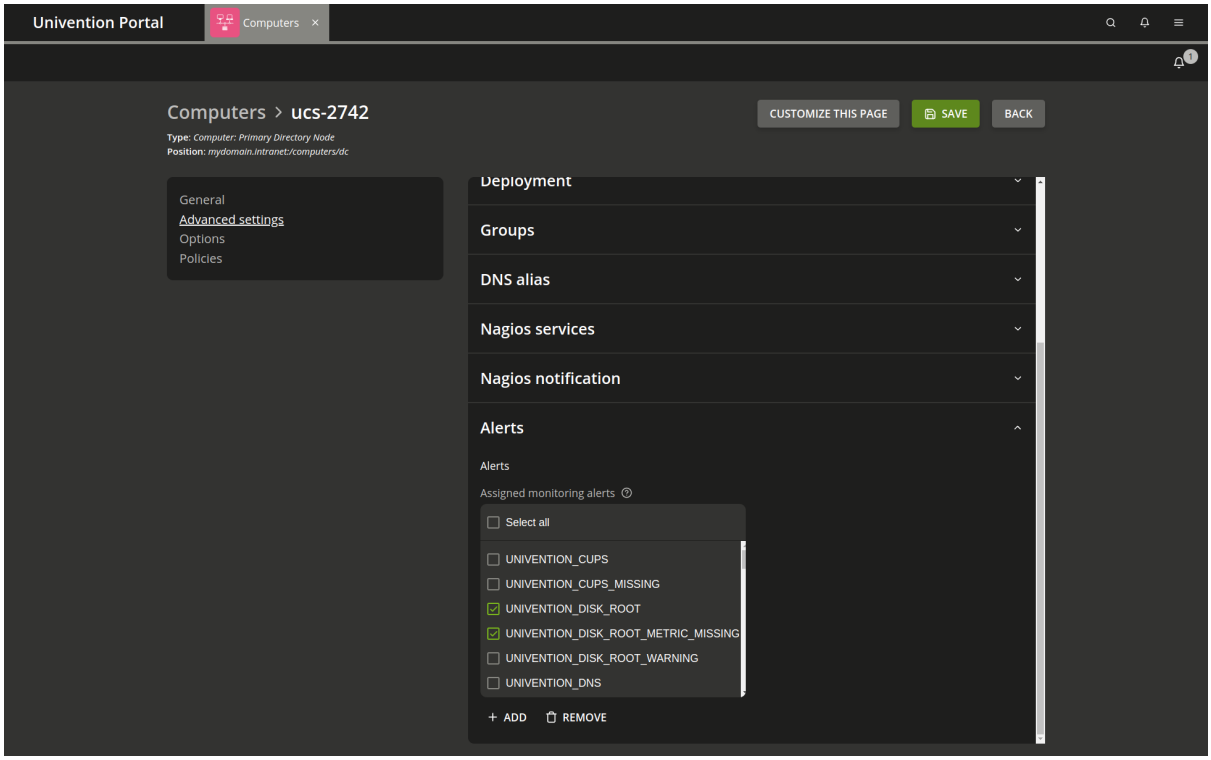


Fig. 15.6: Assigning alert to a host

Table 15.5: *Advanced settings* tab

Attribute	Description
`Assigned monitoring alerts`	Lists all assigned monitoring alerts for the current computer. Add or remove alerts.

Create new alerts

This section describes how to add a custom script to collect new metrics and create new alerts.

As administrator, you can complement the preconfigured alerts supplied with UCS with additional alerts. An alert check script exports metrics about the machine it runs on to *Prometheus*. A *PromQL* query on metrics defines an alert in *Prometheus*. For more information about how to write custom checks, see Querying basis[71].

Copy the custom alert check script into the directory `/usr/share/univention-monitoring-client/scripts/` on the UCS system that shall export the custom metrics. Change the file mode to *executable* with **`chmod a+x PLUGIN`**.

All alert checks delivered by UCS use Python. Custom checks can use Perl, Python, or Shell and don't require any external libraries or programs. All UCS systems always provide the needed interpreters.

In contrast, if the custom alert check uses external programs or libraries, ensure you install them on all UCS systems that use the custom alert check.

The alert check script exports one or multiple metrics by writing them to a text file. It must write valid *Prometheus* metrics into a `.prom` file in the `/var/lib/prometheus/node-exporter/` directory. *Prometheus* imports this file.

You need to configure the custom alert in Univention Management Console, see *Configure monitoring alerts* (page 285). You must enter a Prometheus expression for the metric of the script to the *Query expression* field. To assign the custom alert to UCS systems, see *Assign monitoring alerts to computers* (page 286).

See also:

Prometheus naming conventions
Metric and label naming[72]

Text-based format of a `.prom` file
Exposition formats[73]

15.3 Nagios

With UCS 5.0 support for the Nagios server component has been discontinued. Yet, the systems can still be monitored, e.g. by UCS 4.4 Nagios servers, as described in the UCS 4.4 manual.

15.3.1 Installation

In addition to the standard plugins provided with the installation of the **`univention-nagios-client`** package, additional plugins can be subsequently installed with the following packages:

- **`univention-nagios-raid`**: Monitoring of the software RAID status
- **`univention-nagios-smart`**: Test of the S.M.A.R.T. status of hard drives
- **`univention-nagios-opsi`**: Test of software distribution OPSI
- **`univention-nagios-ad-connector`**: Test of the AD Connector

Some of the packages are automatically set up during installation of the respective services. For example, if the UCS AD connector is set up, the monitoring plugin is included automatically.

[71] https://prometheus.io/docs/prometheus/latest/querying/basics/
[72] https://prometheus.io/docs/practices/naming/
[73] https://prometheus.io/docs/instrumenting/exposition_formats/

15.3.2 Preconfigured Nagios checks

During the installation, basic Nagios tests are set up automatically for UCS systems.

Table 15.6: Preconfigured Nagios checks

Nagios service	Description
UNIVENTION_PING	Tests the availability of the monitored UCS system with the command **ping**. By default an error status is attained if the response time exceeds 50 ms or 100 ms or package losses of 20% or 40% occur.
UNIVENTION_DISK_ROOT	Monitors how full the / partition is. An error status is raised if the remaining free space falls below 25% or 10% by default.
UNIVENTION_DNS	Tests the function of the local DNS server and the accessibility of the public DNS server by querying the hostname www.univention.de. If no DNS forwarder is defined for the UCS domain, this request fails. In this case, www.univention.de can be replaced with the FQDN of the Primary Directory Node for example, in order to test the function of the name resolution.
UNIVENTION_LDAP	Monitors the LDAP server running on UCS Directory Nodes.
UNIVENTION_LOAD	Monitors the system load.
UNIVENTION_NTP	Requests the time from the NTP service on the monitored UCS system. If this deviates by more than 60 or 120 seconds, the error status is attained.
UNIVENTION_SMTP	Tests the mail server.
UNIVENTION_SSL	Tests the remaining validity period of the UCS SSL certificates. This plugin is only suitable for Primary Directory Node and Backup Directory Node systems.
UNIVENTION_SWAP	Monitors the utilization of the swap partition. An error status is raised if the remaining free space falls below the threshold (40% or 20% by default).
UNIVENTION_REPLICATION	Monitors the status of the LDAP replication and recognizes the creation of a failed.ldif file and the standstill of the replication and warns of large differences between the transaction IDs.
UNIVENTION_NSCD	Tests the availability of the name server cache daemon (NSCD). If there is no NSCD process running, a CRITICAL event is triggered; if more than one process is running, a WARNING.
UNIVENTION_WINBIND	Tests the availability of the Winbind service. If no process is running, a CRITICAL event is triggered.
UNIVENTION_SMBD	Tests the availability of the Samba service. If no process is running, a CRITICAL event is triggered.
UNIVENTION_NMBD	Tests the availability of the NMBD service, which is responsible for the NetBIOS service in Samba. If no process is running, a CRITICAL event is triggered.
UNIVENTION_JOINSTATUS	Tests the join status of a system. If a system has yet to join, a CRITICAL event is triggered; if non-run join scripts are available, a WARNING event is returned.
UNIVENTION_KPASSWDD	Tests the availability of the Kerberos password service (only available on Primary/Backup Directory Nodes). If fewer or more than one process is running, a CRITICAL event is triggered.
UNIVENTION_CUPS	Monitors the CUPS daemon. If there is no **cupsd** process running or the web interface on port 631 is not accessible, the CRITICAL status is returned.
UNIVENTION_SQUID	Monitors the Squid proxy. If no squid process is running or the Squid proxy is not accessible, the CRITICAL status is returned.

The following Nagios services are only available on the respective Nagios client once additional packages have been installed (see *Installation* (page 287)):

Table 15.7: Additional Nagios checks

Nagios service	Description
UNIVENTION_OPSI	Monitors the OPSI Daemon. If no OPSI process is running or the OPSI proxy is not accessible, the CRITICAL status is returned.
UNIVENTION_SMART_SDA	Tests the S.M.A.R.T. status of the hard drive /dev/sda. Corresponding Nagios services exist for the hard drives sdb, sdc and sdd.
UNIVENTION_RAID	Tests the status of the software RAID via /proc/mdadm and returns CRITICAL if one of the hard drives in the RAID association has failed or WARNING if a recovery procedure is in progress.
UNIVENTION_ADCONNECTOR	Checks the status of the AD connector: • If no connector process is running, CRITICAL is reported. • If more than one process is running per connector instance, a WARNING is given. • If rejects occur, a WARNING is given. • If the AD server cannot be reached, a CRITICAL status occurs. The plugin can also be used in multi-connector instances; the name of the instance must be passed on as a parameter.

CHAPTER

SIXTEEN

APPENDIX

16.1 Univention Configuration Registry Variables

This appendix lists the Univention Configuration Registry Variables mentioned in the document.

auth/faillog

Configures the automatic locking of users after failed login attempts in the PAM stack. To activate, set the value to `yes`. For more information, see *PAM-Stack* (page 127).

auth/faillog/limit

Configures the upper limit of failed login attempts for a user account lockout. For more information, see *PAM-Stack* (page 127).

auth/faillog/lock_global

Configure on Primary Directory Node and Backup Directory Node to create a failed login account lockout globally and store it in the LDAP directory. For more information, see *PAM-Stack* (page 127).

auth/faillog/root

To make the user account `root` subject of the PAM stack account lockout, set the value to `yes`. It defaults to `no`. For more information, see *PAM-Stack* (page 127).

auth/faillog/unlock_time

Configure a time interval to unlock an account lockout. The value is defined in seconds. The value `0` resets the lock immediately. For more information, see *PAM-Stack* (page 127).

auth/sshd/user/root

To prohibit SSH login for the user `root` completely, set the value `no`. For more information, see *SSH login to systems* (page 168).

backup/clean/max_age

Defines how long a UCS system keeps old backup files of the LDAP data. Allowed values are integer numbers and they define days. The system doesn't delete backup files when the variable is not set. See *Daily backup of LDAP data* (page 40).

connector/ad/ldap/binddn

Configures the LDAP DN of a privileged replication user. For more information, see *UCS as a member of an Active Directory domain* (page 181) and *Changing the AD access password* (page 188).

connector/ad/ldap/bindpw

Configures the password of a privileged replication user. For more information, see *UCS as a member of an Active Directory domain* (page 181) and *Changing the AD access password* (page 188).

connector/ad/ldap/ssl

To deactivate encrypted communication between the UCS system and Active Directory set the value to `no`. For more information, see *Importing the SSL certificate of the Active Directory* (page 185).

connector/ad/mapping/group/language

Configures the mapping for group name conversion in anglophone AD domains. For more information, see *Groups* (page 194).

connector/ad/poll/sleep

Configures the interval to poll for changes in the AD domain. The default is `5` seconds. For more information, see *Setup of the UCS AD connector* (page 183).

connector/ad/retryrejected

Configures the number of cycles that the UCS AD Connector attempts to synchronize an object from the AD domain when it can't be synchronized. The default value is `10` cycles. For more information, see *Setup of the UCS AD connector* (page 183).

cups/cups-pdf/anonymous

Configures the target directory for the *Generic CUPS-PDF Printer* for anonymous print jobs. It defaults to the value `/var/spool/cups-pdf/`. For more information, see *Generating PDF documents from print jobs* (page 255).

cups/cups-pdf/cleanup/enabled

To cleanup outdated print jobs of the *Generic CUPS-PDF Printer*, set the value to `true`. For the storage time, see `cups/cups-pdf/cleanup/keep` (page 292). For more information, see *Generating PDF documents from print jobs* (page 255).

cups/cups-pdf/cleanup/keep

Configures the storage time in days for PDF files from the *Generic CUPS-PDF Printer*. For more information, see *Generating PDF documents from print jobs* (page 255).

cups/cups-pdf/directory

Configures the target directory for the *Generic CUPS-PDF Printer*. It defaults to the value `/var/spool/cups-pdf/%U` and uses a different directory for each user. For more information, see *Generating PDF documents from print jobs* (page 255).

cups/errorpolicy

To automatically retry unsuccessful print jobs every 30 seconds, set the value to `retry-job`. For more information, see *Setting the local configuration properties of a print server* (page 250).

cups/include/local

To include configuration from `/etc/cups/cupsd.local.conf`, set the value to `true`. For more information, see *Setting the local configuration properties of a print server* (page 250).

cups/server

Defines the print server to be used by a UCS system. For more information, see *Configuration of the print server in use* (page 165).

directory/manager/blocklist/cleanup/cron

This variable determines how often UDM searches and removes the expired block list entries. The value follows the *cron syntax* (page 167) for the time definition. The default value is daily at 8:00 in the morning. For more information, see *Expired block list entries* (page 132).

directory/manager/blocklist/enabled

Activates the management of block list entries in UDM. Default value is `false`. For information about how to activate, see *Activate block lists* (page 131).

directory/manager/templates/alphanum/whitelist

Define an allowlist of characters that are not removed by the `:alphanum` option for the value definition in user templates. For more information, see *User templates* (page 129).

directory/manager/user_group/uniqueness

Controls if UCS prevents users with the same username as existing groups. To deactivate the check for uniqueness, set the value to `false`. For more information see Table 6.1.

directory/manager/web/modules/computers/computer/wizard/disabled

To disable the simplified wizard for computer management, set this variable to `true`. For more information, see *Management of computer accounts via Univention Management Console module* (page 143).

directory/manager/web/modules/groups/group/checks/circular_dependency

Controls the check for circular dependencies regarding nested groups. To disable, set the value to `no`. For more information, see *Group nesting with groups in groups* (page 139).

directory/manager/web/modules/users/user/wizard/disabled

Deactivates the simplified wizard to create users when the value is set to `true`. In the default setting the wizard is activated. For more information see *User management through Univention Management Console module* (page 106).

directory/reports/logo

Defines the path and name to an image file for usage as logo in a Univention Directory report PDF file. For more information see *Adjustment/expansion of Univention Directory Reports* (page 89).

dns/allow/transfer

To deactivate the DNS zone transfer when using the OpenLDAP backend, set the value to `none`. For more information, see *Configuration of zone transfers* (page 214).

dns/backend

Configures the DNS backend. For more information, see *Configuration of the data backend* (page 213).

dns/debug/level

Configures the debug level for BIND. For more information, see *Configuration of BIND debug output* (page 213).

dns/dlz/debug/level

Configures the debug level for the Samba DNS backend. For more information, see *Configuration of BIND debug output* (page 213).

dns/forwarder1

Defines the first *external DNS server*. For more information, see *Configuring the name servers* (page 153).

dns/forwarder2

Defines the second *external DNS server*. For more information, see *Configuring the name servers* (page 153).

dns/forwarder3

Defines the third *external DNS server*. For more information, see *Configuring the name servers* (page 153).

fetchmail/autostart

Controls the automatic start of Fetchmail. To disable Fetchmail, set the value to `false`. For more information, see *Integration of Fetchmail for retrieving mail from external mailboxes* (page 268).

freeradius/auth/helper/ntlm/debug

Configures the debug level or verbosity for logging messages of FreeRADIUS. For more information, see *Debugging* (page 235).

freeradius/conf/allow-mac-address-authentication

Configures if Radius allows the MAC address as username and password for 802.1X authentication. Default value is `false`. For more information, see *MAC Authentication Bypass with computer objects* (page 230).

freeradius/conf/mac-addr-regexp

Configures the regular expression for the MAC address for the Radius server. The regular expression must contain six groups, each group representing a byte of the MAC address.

The default value is the regular expression: `([0-9a-f]{2})[^0-9a-f]?([0-9a-f]{2})[^0-9a-f]?([0-9a-f]{2})[^0-9a-f]?([0-9a-f]{2})[^0-9a-f]?([0-9a-f]{2})[^0-9a-f]?([0-9a-f]{2})`

New in version 5.0-6-erratum-…: With UCS 5.0 erratum 1011[74] the Radius server can handle different formats of the MAC addresses for the username when using MAB.

For more information and effects, see *MAC Authentication Bypass with computer objects* (page 230).

freeradius/vlan-id

Configures the fallback value for VLAN ID for users that aren't member of a group with a VLAN ID. For more information, see *VLAN IDs* (page 234).

gateway

Configures the IPv4 network gateway. For more information, see *Configuration of IPv4 addresses* (page 152).

google-apps/attributes/anonymize

Configures the LDAP attributes of a user account that Google Apps for Work Connector synchronizes, but fills with random data. The value is a comma-separated list of LDAP attributes. For more information, see *Configuration* (page 207).

google-apps/attributes/mapping/.*

Defines a mapping of UCS LDAP attributes of a user account for synchronization to Google Apps attributes. The default settings usually suffice most environment needs. For more information, see *Configuration* (page 207).

google-apps/attributes/never

Configures the LDAP attributes of a user account that the Google Apps for Work Connector never synchronizes, even if mentioned in `google-apps/attributes/mapping/.*` (page 294) or `google-apps/attributes/anonymize` (page 294). The value is a comma-separated list of LDAP attributes. For more information, see *Configuration* (page 207).

google-apps/debug/werror

Configure additional debug error messages for the Google Apps for Work. For more information, see *Troubleshooting/Debugging* (page 209).

google-apps/groups/sync

Enables the synchronization of groups of the Google Apps for Work user groups with the value `yes`. For more information, see *Configuration* (page 207).

groups/default/domainadmins

Configures the default group name for the domain administrator group. The value might be changed during an AD Takeover. For more information, see *Domain migration* (page 197).

grub/append

Defines Linux kernel boot options that the GRUB boot loader passes to the Linux kernel for system boot. For more information, see *GRUB boot manager* (page 150).

grub/bootsplash

To deactivate the splash screen during system boot, set the value to `nosplash`. For more information, see *GRUB boot manager* (page 150).

grub/gfxmode

Defines screen size and color depth for the GRUB boot menu. For more information, see *GRUB boot manager* (page 150).

grub/timeout

Defines the waiting period in seconds in the GRUB boot menu. During this waiting time alternative boot menu entries can be selected. The default value is `5` seconds. For more information, see *GRUB boot manager* (page 150).

grub/xenhopt

Defines options that are passed to the Xen hypervisor. For more information, see *GRUB boot manager* (page 150).

[74] https://errata.software-univention.de/#/?erratum=5.0x1011

interfaces/ethX/address

Defines the network IPv4 address for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv4 addresses* (page 152).

interfaces/ethX/netmask

Defines the network mask for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv4 addresses* (page 152).

interfaces/ethX/type

Defines the network interface type for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv4 addresses* (page 152).

interfaces/ethX_Y/setting

Defines an additional virtual interface. Replace `X` and `Y` with the actual value for the interface. For more information, see *Configuration of IPv4 addresses* (page 152).

interfaces/ethX/ipv6/address

Defines the network IPv6 address for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv6 addresses* (page 152).

interfaces/ethX/ipv6/prefix

Defines the network IPv6 prefix for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv6 addresses* (page 152).

interfaces/ethX/ipv6/acceptRA

Activates stateless address autoconfiguration (SLAAC) for the interface `ethX`. Replace `X` with the actual value for the interface. For more information, see *Configuration of IPv6 addresses* (page 152).

ipv6/gateway

Configures the IPv4 network gateway. For more information, see *Configuration of IPv6 addresses* (page 152).

kerberos/adminserver

Defines the system that provides the Kerberos admin server. See *Kerberos admin server* (page 44).

kerberos/kdc

Contains the reference to the KDC. Typically, a UCS system selects the KDC to be used from a DNS service record. With this variable administrators can configure an alternative KDC.

kerberos/realm

Contains the name of the Kerberos realm. See *Kerberos* (page 44).

kernel/blacklist

Defines additional Linux kernel modules that need to be loaded during system boot. Single items must be separated with a semicolon (`;`). For more information, see *Hardware drivers / kernel modules* (page 149).

kernel/modules

Defines Linux kernel modules that must not be loaded during system boot. Single items must be separated with a semicolon (`;`). For more information, see *Hardware drivers / kernel modules* (page 149).

ldap/database/internal/acl/blocklists/groups/read

List of group distinguished names (DN) that have read access to all objects under the container `cn=blocklists` in the internal database. For more information, see *LDAP ACLs for block lists* (page 133).

ldap/database/internal/acl/blocklists/groups/write

List of group distinguished names (DN) that have write access to all objects under the container `cn=blocklists` in the internal database. For more information, see *LDAP ACLs for block lists* (page 133).

ldap/acl/read/anonymous

Controls if the LDAP server allows anonymous access to the LDAP directory. In the default configuration the LDAP server doesn't allow anonymous access to the LDAP directory.

ldap/acl/read/ips

A list of IP addresses for which the LDAP server allows anonymous access. See *Access control for the LDAP directory* (page 38).

ldap/acl/nestedgroups

Controls if nested groups are allowed. Per default nested groups are activated. See *Access control for the LDAP directory* (page 38).

ldap/acl/user/passwordreset/accesslist/groups/dn

Use a different group from the default `User Password Admins` group to reset user passwords. The value is a distinguished name (DN) to a user group. See *Delegation of the privilege to reset user passwords* (page 39).

ldap/acl/user/passwordreset/attributes

If users that are allowed to change other users' passwords need access to additional LDAP attributes needed for the password change, configure them in this variable. For more information, see *Delegation of the privilege to reset user passwords* (page 39).

ldap/acl/user/passwordreset/protected/uid

Configures users with their user id to exclude them from user password resets by administrators allowed to change user passwords. Separate multiple values with a comma. For more information, see *Delegation of the privilege to reset user passwords* (page 39).

ldap/acl/user/passwordreset/protected/gid

Configures groups with their group id to exclude them from user password resets by administrators allowed to change user passwords. Separate multiple values with a comma. For more information, see *Delegation of the privilege to reset user passwords* (page 39).

ldap/idletimeout

Configures a time period in seconds after which the LDAP connection is cut off on the server side. See *Timeout for inactive LDAP connections* (page 38).

ldap/logging/exclude1

Exclude individual areas of the directory service from logging. See *Audit-proof logging of LDAP changes* (page 37).

ldap/logging/excludeN

See *ldap/logging/exclude1* (page 296).

ldap/logging/id-prefix

Adds the transaction ID of an entry to the directory log. Possible values are the default `yes` and `no`. See *Audit-proof logging of LDAP changes* (page 37).

ldap/master

Contains the FQDN of the domain's Primary Directory Node system.

ldap/overlay/lastbind

To activate the `lastbind` overlay module for the LDAP server, set the value to `yes`. For more information, see *Overlay module for recording an account's last successful LDAP bind* (page 130).

ldap/overlay/lastbind/precision

Configures the time in seconds that has to pass before the `authTimestamp` is updated again by the `lastbind` overlay. For more information, see *Overlay module for recording an account's last successful LDAP bind* (page 130).

ldap/overlay/memberof/memberof

Configures the attribute at user objects that shows the group membership. Default value is `memberOf`. For more information, see *Overlay module for displaying the group information on user objects* (page 141).

ldap/policy/cron

Time interval to write profile based UCR variables to a UCS system. The default value is one hour. For more information, see *Policy-based configuration of UCR variables* (page 160).

ldap/ppolicy

To enable automatic account locking, set the value to `yes`. Also set `ldap/ppolicy/enabled` (page 297). For more information, see *OpenLDAP* (page 128).

ldap/ppolicy/enabled

To enable automatic account locking, set the value to `yes`. Also set `ldap/ppolicy` (page 296). For more information, see *OpenLDAP* (page 128).

ldap/pw-bcrypt

Activates **bcrypt** as password hashing method when set to `true`. See *Password hashes in the directory service* (page 44).

ldap/server/addition

Additional LDAP server a UCS system can query for information in the directory service.

ldap/server/name

The LDAP server the system queries for information in the directory service.

listener/debug/level

Defines the detail level for log messages of the listener to `/var/log/univention/listener.log`. The possible values are from 0 (only error messages) to 4 (all status messages). Once the debug level has been changed, the Univention Directory Listener must be restarted.

listener/shares/rename

Contents of existing share directories are moved, when the path to a share is modified and the value is set to `yes`. For more information, see Table 12.1 in *Shares UMC module - General tab* (page 239).

local/repository

Activates and deactivates the local repository. For more information see *Creating and updating a local repository* (page 99).

logrotate/compress

Controls, if rotated log files are zipped with **gzip**. For more information, see *Log files* (page 165).

log/rotate/weeks

Configures the log file rotation interval on a UCS system in weeks. The default value is `12` weeks. For more information, see *Log files* (page 165).

logrotate/rotates

Configures the log file rotation according to the file size, for example `size 50M`. For more information, see *Log files* (page 165).

machine/password/length

Define the length for the computer password, also called *machin secret*. Default value is `20`. For more information, see *Management of computer accounts via Univention Management Console module* (page 143).

mail/antispam/bodysizelimit

Configures the size of emails that are scanned for Spam by SpamAssassin. The default value is 300 kilobytes. For more information, see *Spam detection and filtering* (page 267).

mail/antispam/learndaily

Configures the evaluation of ham emails in the ham folder for daily evaluation. The evaluation is activate per default. For more information, see *Spam detection and filtering* (page 267).

mail/antispam/requiredhits

Configures the threshold in points when an email is classified as spam. The default value is `5`. For more information, see *Spam detection and filtering* (page 267).

mail/antivir

To deactivate virus and malware detection for incoming and outgoing emails, set the value to `no`. For more information, see *Identification of viruses and malware* (page 267).

mail/antivir/spam

Configures, if spam filtering is running. To deactivate spam filtering, set the value to `no`. For more information, see *Identification of viruses and malware* (page 267).

mail/archivefolder

Configures Postfix to send all incoming and outgoing emails as blind copy to this email address for archive purposes. The variable isn't set per default. For more information, see *Configuration of a blind carbon copy for mail archiving solutions* (page 271).

mail/dovecot/auth/cache_ttl

Configures the expiration time of the authentication cache in Dovecot for the mail service. For more information, see *Assignment of email addresses to users* (page 262).

mail/dovecot/auth/cache_negative_ttl

Configures the expiration time of the authentication cache in Dovecot for the mail service. For more information, see *Assignment of email addresses to users* (page 262).

mail/dovecot/folder/ham

Configures the name of the folder for emails that SpamAssissin considers as *ham*. The default value is `Ham`. For more information, see *Spam detection and filtering* (page 267).

mail/dovecot/folder/Spam

Configures the name of the folder where SpamAssissin moves emails classified as spam. The default value is `Spam`. For more information, see *Spam detection and filtering* (page 267).

mail/dovecot/imap

Controls the IMAP protocol service in the Dovecot IMAP service. To deactivate access to emails through IMAP, set the value to `no`. For more information, see *Mail services* (page 261).

mail/dovecot/limits

Configures different connection limits for the Dovecot service. For more information, see *Connection limits* (page 274).

mail/dovecot/location/separate_index

Configures the Dovecot service to use an index separated from the email message storage location. To activate the separate index, set the value to `yes`. Dovecot writes the index to `/var/lib/dovecot/index/`. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/mailbox/rename

Configures how the Dovecot services reacts on changes of the primary email address. The default value is `yes` and it changes the name of the user's IMAP mailbox. For more information about the values, see *Handling of mailboxes during email changes and the deletion of user accounts* (page 273).

Shared folders are not renamed. For more information, see *Management of shared IMAP folders* (page 264).

mail/dovecot/mailbox/delete

Configures the deletion of an IMAP mailbox. The default value is `no` and keeps the mailbox. For more information, see *Handling of mailboxes during email changes and the deletion of user accounts* (page 273).

The value also affects shared IMAP folders. For more information, see *Management of shared IMAP folders* (page 264).

mail/dovecot/pop3

Controls the POP3 protocol service in the Dovecot IMAP service. To deactivate access to emails through POP3, set the value to `no`. For more information, see *Mail services* (page 261).

mail/dovecot/process/lock_method

Controls the lock method for *lockd*. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/process/mail_nfs_index

Configures the Dovecot service to flush NFS caches after writing index files when set to `yes`. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/process/mail_nfs_storage

Configures the Dovecot service to flush NFS caches when set to `yes`. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/process/mmap_disable

Allows mail storage on NFS. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/process/dotlock_use_excl

Allows mail storage on NFS. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/process/mail_fsync

Allows mail storage on NFS. For more information, see *Mail storage on NFS* (page 274).

mail/dovecot/quota/warning/subject

Configures the subject for the email to the user that exceeds the configured quota limit. For more information, see *Mail quota* (page 266).

mail/dovecot/quota/warning/text

Configures the email text body for the email to the user that exceeds the configured quota limit. Percentage values can have different texts. For example, to configure a text for 50% of the quota, set `mail/dovecot/quota/warning/text/50=`*`your text`*.

For more information, see *Mail quota* (page 266).

mail/hosteddomains

Configures the mail domains managed by UCS. For more information, see *Management of mail domains* (page 262).

mail/messagesizelimit

Configures the maximum size of an email in bytes for incoming and outgoing emails. The default setting is `10240000` bytes. For more information, see *Configuration of the maximum mail size* (page 271).

mail/postfix/mastercf/options/smtp/smtpd_sasl_auth_enable

To enable authentication for the submission of emails on port 25, set the value to `yes`. For more information, see *Configuration of SMTP ports* (page 271).

mail/postfix/policy/listfilter

To restrict the circle of persons who can send emails to mailing lists, set the value to `yes` and restart the Postfix service. For more information, see *Management of mailing lists* (page 263) and *Management of mail groups* (page 264).

mail/postfix/postscreen/

A prefix of variables to configure **postscreen**. For more information, see *Configuration of additional checks* (page 271).

mail/postfix/postscreen/enabled

To activate postscreen for eligibility checks on incoming emails, set the value to `yes`. For more information, see *Configuration of additional checks* (page 271).

mail/postfix/smtpd/restrictions/recipient

Configures DNS-based Blackhole List (DNSBL) for Postfix in the format `mail/postfix/smtpd/restrictions/recipient/`*`SEQUENCE=RULE`*.

For example:
`mail/postfix/smtpd/restrictions/recipient/80="reject_rbl_client ix.dnsbl.manitu.net"`.

For more information, see *Identification of Spam sources with DNS-based Blackhole Lists* (page 268).

mail/postfix/softbounce

To never return emails after a mail bounce, set the value to `yes`. For more information, see *Configuration of soft bounces* (page 271).

mail/postfix/tls/client/level

For more information, see *Configuration of a relay host for sending the emails* (page 269).

mail/relayauth

If authentication for the mail relay is needed, set the value to `yes` and add the credentials to `/etc/postfix/smtp_auth`. For more information, see *Configuration of a relay host for sending the emails* (page 269).

mail/relayhost

Configures the fully qualified domain name (FQDN) of a mail relay server. For more information, see *Configuration of a relay host for sending the emails* (page 269).

nameserver1

Defines the first *Domain DNS Server*. For more information, see *Configuring the name servers* (page 153).

nameserver2

Defines the second *Domain DNS Server*. For more information, see *Configuring the name servers* (page 153).

nameserver3

Defines the third *Domain DNS Server*. For more information, see *Configuring the name servers* (page 153).

notifier/debug/level

Defines the detail level for log messages of the notifier to `/var/log/univention/notifier.log`. The possible values are from 0 (only error messages) to 4 (all status messages). Once the debug level has been changed, the Univention Directory Notifier must be restarted.

nscd/debug/level

Defines the detail level for log messages of the NSCD. For more information, see *Name service cache daemon* (page 168).

nscd/group/maxdbsize

Configures the hash table size of the NSCD for groups. For more information, see *Name service cache daemon* (page 168).

nscd/group/positive_time_to_live

Configures the time that a resolved group is kept in the cache of NSCD. The default is one hour in seconds (`3600`). For more information, see *Name service cache daemon* (page 168).

nscd/hosts/maxdbsize

Configures the hash table size of the NSCD for hosts. The default value is `6007`. For more information, see *Name service cache daemon* (page 168).

nscd/hosts/positive_time_to_live

Configures the time that a resolved hostname is kept in the cache of NSCD. The default is one hour in seconds (`3600`). For more information, see *Name service cache daemon* (page 168).

nscd/passwd/maxdbsize

Configures the hash table size of the NSCD for usernames. The default value is `6007`. For more information, see *Name service cache daemon* (page 168).

nscd/passwd/positive_time_to_live

Configures the time that a resolved username is kept in the cache of NSCD. The default is ten minutes in seconds (`600`). For more information, see *Name service cache daemon* (page 168).

nscd/threads

Configures the number of threads that NSCD uses. Default value is `5`. For more information, see *Name service cache daemon* (page 168).

nss/group/cachefile/check_member

When activated with `true`, the cron job script for exporting the local group cache also checks, if the group members are still present in the LDAP directory. For more information, see *Local group cache* (page 139).

nss/group/cachefile/invalidate_interval

Defines the invalidation interval to control when the local group cache is considered invalid and a new export is run. For more information, see *Local group cache* (page 139).

nss/group/cachefile/invalidate_on_changes

Activates or deactivates the listener to invalidate the local group cache. To activate the listener, set the value to `true`. Else, set it to `false`. For more information, see *Local group cache* (page 139).

nssldap/bindpolicy

Controls the measures that the UCS system takes when the LDAP server cannot be reached. See *Name Service Switch / LDAP NSS module* (page 40).

ntp/signed

The NTP server replies with requests that are signed by Samba/AD when the value is set to `yes`. For more information, see *Configuring the time zone / time synchronization* (page 169).

office365/adconnection/wizard

Defines the Azure AD connection alias that is used by the next run of the Microsoft 365 Configuration Wizard. For more information, see *Synchronization of Users in multiple Azure Active Directories* (page 206).

office365/attributes/anonymize

Configures the LDAP attributes of a user account that the Microsoft 365 connector synchronizes, but fills with random data. The value is a comma-separated list of LDAP attributes. For more information, see *Users* (page 204).

office365/attributes/mapping/.*

Defines a mapping of UCS LDAP attributes of a user account for synchronization to Azure attributes. The default settings usually suffice most environment needs. For more information, see *Users* (page 204).

office365/attributes/never

Configures the LDAP attributes of a user account that the Microsoft 365 connector never synchronizes, even if mentioned in *office365/attributes/sync* (page 301) or *office365/attributes/anonymize* (page 301). The value is a comma-separated list of LDAP attributes. For more information, see *Users* (page 204).

office365/attributes/static/.*

Configures LDAP attributes for synchronization with predefined values. For more information, see *Users* (page 204).

office365/attributes/sync

Configures the LDAP attributes of a user account that the Microsoft 365 connector synchronizes. The value is a comma-separated list of LDAP attributes. For more information, see *Users* (page 204).

office365/attributes/usageLocation

Configures the default country for the user in Microsoft 365. Values are 2-character abbreviations for countries. For more information, see *Users* (page 204).

office365/debug/werror

Configure additional debug error for the Microsoft 365 connector. For more information, see *Troubleshooting/Debugging* (page 206).

office365/defaultalias

Configures the default connection alias for Microsoft 365 enabled users and groups. For more information, see *Synchronization of Users in multiple Azure Active Directories* (page 206).

office365/groups/sync

Enables the synchronization of groups of the Microsoft 365 users. To use teams, set the value to `yes`. For more information, see *Teams* (page 206).

password/hashing/bcrypt

Activates **bcrypt** as password hashing method when set to `true`. See *Password hashes in the directory service* (page 44).

password/hashing/bcrypt/cost_factor

Defines the **bcrypt** cost factor and defaults to `12`. See *Password hashes in the directory service* (page 44).

password/hashing/bcrypt/prefix

Defines the **bcrypt** prefix and defaults to `2b`. See *Password hashes in the directory service* (page 44).

password/hashing/method

Defines the hashing method for the password hashes. The default is `SHA-512`. See *Password hashes in the directory service* (page 44).

password/quality/credit/digits

Defines the minimum required number of digits for a password. For more information, see *User password management* (page 115).

password/quality/credit/lower

Defines the minimum required number of lowercase letters in the password. For more information, see *User password management* (page 115).

password/quality/credit/other

Defines the minimum required number of characters in the password which are neither letters nor digits. For more information, see *User password management* (page 115).

password/quality/credit/upper

Defines the minimum required number of uppercase letters in the password. For more information, see *User password management* (page 115).

password/quality/forbidden/chars

Defines the characters and digits not allowed for passwords. For more information, see *User password management* (page 115).

password/quality/length/min

Defines the minimum required default length for a password on a per UCS system basis for users not subject to a UDM password policy. The value `yes` applies checks from the **python-cracklib**. The value `sufficient` doesn't include **python-cracklib** checks. For more information, see *User password management* (page 115).

password/quality/mspolicy

Defines the standard Microsoft password complexity criteria.

The values `yes`, `1` or `true` activate the standard Microsoft password complexity criteria in addition to the other criteria validated with **python-cracklib**.

The value `sufficient` only applies the standard Microsoft password complexity criteria without **python-cracklib**.

The default value is unset and corresponds to the value `false`.

For more information, see *User password management* (page 115).

password/quality/required/chars

Defines individual characters that are required for passwords. For more information, see *User password management* (page 115).

pkgdb/scan

Controls if a UCS system stores installation processes in the software monitor. To turn it off, set the value `no`. For more information see *Central monitoring of software installation statuses with the software monitor* (page 103).

portal/auth-mode

Defines the authentication mode for the UCS portal. Set it to `saml`, if you want to activate SAML for single sign-on login. For more information see *Login* (page 57).

portal/default-dn

Defines the LDAP DN of the portal object that holds the data for the portal. After changing the variable value, run **univention-portal update**. For more information, see *UCS portal page* (page 65).

proxy/http

Defines the HTTP proxy server on the UCS host system. For more information, see *Proxy access configuration* (page 155).

proxy/https

Defines the HTTPS proxy server on the UCS host system. For more information, see *Proxy access configuration* (page 155).

proxy/no_proxy

Defines a list of domains that are not used over a HTTP proxy. Entries are separated by commas. For more information, see *Proxy access configuration* (page 155).

quota/logfile

To log the activation of quotas to a file, specify the file in this variable. For more information, see *Evaluation of quota during login* (page 247).

quota/userdefault

To disable the evaluation of user quota during login, set the value to `no`. For more information, see *Evaluation of quota during login* (page 247).

radius/mac/whitelisting

To only allow specific network devices access to a network through RADIUS, set the value to `true`. For more information, see *MAC filtering* (page 230).

radius/use-service-specific-password

To use a dedicated user password for RADIUS instead of the domain password, set the value to `true`. For more information, see *Service specific password* (page 230).

repository/mirror/server

Defines another repository server as source for the local mirror. Default value: `updates.software-univention.de`. For more information see *Creating and updating a local repository* (page 99).

repository/online/component/.*/unmaintained

`DEPRECATED!` Defines how to deal with unmaintained packages from additional repositories. To activate, set the value to `yes`. For more information see *Configuration of the repository server for updates and package installations* (page 98).

repository/online/server

The repository server used to check for updates and download packages. Default value: `updates.software-univention.de`. For more information see *Configuration via Univention Configuration Registry* (page 99).

samba/enable-msdfs

To enable Microsoft Distributed File System (MSDFS), set the value to `yes` and restart Samba. For more information, see *Support for MSDFS* (page 245).

samba/max/protocol

Configures the file service protocol that Samba uses on UCS. The allowed values `NT1`, `SMB2`, and `SMB3`. For more information, see *File services* (page 172).

samba/spoolss/architecture

Defines the system architecture for the print spooler in Samba. Set the values to `Windows x64` when your environment contains 64-bit version of Microsoft Windows. For more information, see *Mounting of print shares in Windows clients* (page 255).

samba4/sysvol/sync/cron

Configures the synchronization time interval between Samba/AD domain controllers for the SYSVOL share. Default value is five minutes. For more information, see *Synchronization of the SYSVOL share* (page 173).

saml/idp/authsource

Allows Kerberos authentication at the SAML identity provider. Change to `univention-negotiate` to activate. The default is `univention-ldap`. For more information, see *SAML identity provider* (page 45).

saml/idp/entityID/supplement/[identifier]

Activates additional local identity providers for SAML on a UCS system serving as UCS Identity provider. To activate set the value to `true`. For more information see *Extended Configuration* (page 48).

saml/idp/negotiate/filter-subnets

Allows to restrict the Kerberos authentication at the SAML identity provider to certain IP subnetworks in the CIDR notation[75], for example `127.0.0.0/16,192.168.0.0/16`. For more information, see *SAML identity provider* (page 45).

saml/idp/selfservice/account-verification/error-descr

Configures the error message description text for the **Self Service**. The text shows up for users that login through SSO with an unverified and self registered user account. For more information, see *Account verification* (page 124).

saml/idp/selfservice/account-verification/error-title

Configures the error message title for the **Self Service**. The title shows up for users that login through SSO with an unverified and self registered user account. For more information, see *Account verification* (page 124).

saml/idp/selfservice/check_email_verification

Controls if the SSO login denies logins from unverified and self registered user accounts. For more information, see *Account verification* (page 124).

security/packetfilter/disabled

To disable Univention firewall, set the value to `true`. For more information, see *Packet filter with Univention Firewall* (page 226).

self-service/backend-server

Defines the UCS system where the backend of the **Self Service** app is installed. For more information, see *Password management via Self Service app* (page 119).

server/password/change

Enables or disables the password rotation on a UCS system. Per default the password rotation is activated. For more information, see *Management of computer accounts via Univention Management Console module* (page 143).

server/password/interval

Defines the interval in days to regenerate the computer account password. The default is set to 21 days. For more information, see *Management of computer accounts via Univention Management Console module* (page 143).

server/role

Contains the name of the UCS system's server role. For more information, see *UCS system roles* (page 33).

squid/auth/allowed_groups

To limit the access to the Squid web proxy, define a list of group names separated by semicolon (`;`). For more information, see *User authentication on the proxy* (page 228).

[75] https://en.wikipedia.org/wiki/Classless_Inter-Domain_Routing

squid/allowfrom

Configures additional networks to allow access to the Squid web proxy. Separate the entries with blank spaces and use the CIDR notation, for example `192.0.2.0/24`. For more information, see *Restriction of access to permitted networks* (page 228).

squid/basicauth

To activate direct authentication for the Squid web proxy against the LDAP server, set the value to `yes` and restart Squid. For more information, see *User authentication on the proxy* (page 228).

squid/cache

To deactivate the caching function of the Squid web proxy, set the value to `no`. For more information, see *Caching of web content* (page 227).

squid/httpport

Configures the port for Squid web proxy, where the daemon listens for incoming connections. The default value is `3128`. For more information, see *Access port* (page 228).

squid/krb5auth

To activate authentication through Kerberos for the Squid web proxy, set the value to `yes` and restart Squid. For more information, see *User authentication on the proxy* (page 228).

squid/ntlmauth

To activate authentication for the Squid web proxy against the NTLM interface, set the value to `yes` and restart Squid. For more information, see *User authentication on the proxy* (page 228).

squid/ntlmauth/keepalive

To deactivate further NTML authentication for subsequent HTML requests to the same website, set the value to `yes`. For more information, see *User authentication on the proxy* (page 228).

squid/webports

Configures the list of permitted ports for the Squid web proxy. Separate entries with blank spaces. For more information, see *Permitted ports* (page 228).

sshd/permitroot

Configures how the SSH daemon permits login for the user `root`. The value `without-password` does not ask for the password interactively. The login requires the public SSH key. For more information, see *SSH login to systems* (page 168).

sshd/port

Configure the port that the SSH daemon uses to listen for connection. The default value is `22`. For more information, see *SSH login to systems* (page 168).

sshd/xforwarding

Configures, if the SSH daemon allows X11 forwarding. Valid values are `yes` and `no`. For more information, see *SSH login to systems* (page 168).

ssl/validity/host

Records the expiry date of the local computer certificate on each UCS system. The value reflects the number of days since the 1970-01-01.

ssl/validity/root

Records the expiry date of the root certificate on each UCS system. The value reflects the number of days since the 1970-01-01.

ssl/validity/warning

Defines the warning period for the expiration check of the SSL/TLS root certificate. The default value is `30` days. See *SSL certificate management* (page 43).

system/stats

Enables or disables the logging of the system status. The default value is `yes`. For more information, see *Logging the system status* (page 166).

system/stats/cron

Configures the run times when **univention-system-stats** is run. The value follows the *cron syntax* (page 167). For more information, see *Logging the system status* (page 166).

timeserver

Configures the first external NTP timeserver. For more information, see *Configuring the time zone / time synchronization* (page 169).

timeserver2

Configures the second external NTP timeserver. For more information, see *Configuring the time zone / time synchronization* (page 169).

timeserver3

Configures the third external NTP timeserver. For more information, see *Configuring the time zone / time synchronization* (page 169).

ucr/check/type

If this option is `yes`, the correctness of type definitions and the type compatibility of values are always checked in Univention Configuration Registry. The setting of the value is aborted for unsuccessful checks. Default is `no`. For more information, see *Setting UCR variables* (page 158).

ucs/web/theme

Select the theme for UCS web interface. The value corresponds to a CSS file under `/usr/share/univention-web/themes/` with the same name without filename extension.

umc/self-service/account-deregistration/enabled

To activate the **Self Service** deregistration, set the variable to `True`. For more information, see *Self deregistration* (page 126).

umc/self-service/account-verification/backend/enabled

Enables or disables the account verification and request of new verification tokens for the **Self Service**. For more information, see *Account verification* (page 124).

users/default/administrator

Configures the default user name for the domain administrator. The value might be changed during an AD Takeover. For more information, see *Domain migration* (page 197).

umc/http/session/timeout

Configures the time out period in seconds for the browser session after which the UCS management system requires a renewed sign in. The default value is `28800` seconds for 8 hours.

umc/web/oidc/enabled

If activated with `true`, the UMC first tries a single sign-on login through OpenID Connect before using the regular login. For more information, refer to *Login* (page 57).

umc/oidc/issuer

Configures the OIDC identity provider for the UMC OIDC authentication. If the variable is unset, the value `https://ucs-sso-ng.ucs.test/realms/ucs` is used. For more information, refer to *Login* (page 57).

umc/oidc/rp/server

Defines the FQDN of the relying party for UMC. If the variable is unset, the FQDN of the UCS system and all IP addresses are used. For more information, refer to *Login* (page 57)

umc/web/sso/enabled

If activated with `true`, the UMC first tries a single sign-on login through SAML before using the regular login. For more information, refer to *Login* (page 57).

16.2 Bibliography

16.3 Indices

The genindex provides direct links to curated content topics. It contains the terms from the word list in bold face.

BIBLIOGRAPHY

[1] *UCS documentation overview*. Univention GmbH, 2021. URL: https://docs.software-univention.de/.

[2] *Extended domain services documentation*. Univention GmbH, 2021. URL: https://docs.software-univention.de/ext-domain/5.0/en/.

[3] *Univention Developer Reference*. Univention GmbH, 2021. URL: https://docs.software-univention.de/developer-reference/5.0/en/.

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

[5] *UCS performance guide*. Univention GmbH, 2021. URL: https://docs.software-univention.de/ext-performance/5.0/en/.

[6] *Extended installation documentation*. Univention GmbH, 2021. URL: https://docs.software-univention.de/ext-installation/5.0/en/.

[7] *Extended Windows integration documentation*. Univention GmbH, 2021. URL: https://docs.software-univention.de/ext-windows/5.0/en/.

[8] *Group Policy ADMX Syntax Reference Guide*. Microsoft, July 2021. URL: https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc753471(v=ws.10)?redirectedfrom=MSDN.

[9] *How to Implement the Central Store for Group Policy Admin Templates, Completely (Hint: Remove Those .ADM files!)*. Microsoft, September 2018. URL: https://techcommunity.microsoft.com/blog/coreinfrastructureandsecurityblog/how-to-implement-the-central-store-for-group-policy-admin-templates-completely-h/255448.

[10] Microsoft, editor. *Windows Server 2003/2003 R2*, chapter WMI filtering using GPMC, pages 18372f. Microsoft, January 2005. URL: https://www.microsoft.com/en-US/download/details.aspx?id=53314.

[11] Mark Heitbrink. *Filtern von Gruppenrichtlinien anhand von Benutzergruppen, WMI und Zielgruppenadressierung*. January 2013. URL: https://www.gruppenrichtlinien.de/artikel/filtern-von-gruppenrichtlinien-anhand-von-benutzergruppen-wmi-und-zielgruppenadressierung/.

[12] *Install the Certification Authority*. Microsoft, July 2021. URL: https://learn.microsoft.com/en-us/windows-server/networking/core-network-guide/cncg/server-certs/install-the-certification-authority.

[13] Cricket Liu and Paul Albitz. *DNS and BIND*, chapter 12 Reading BIND Debugging Output, pages 502. O'Reilly, 3rd edition, September 1998. URL: https://www.diablotin.com/librairie/networking/dnsbind/ch12_01.htm.

[14] *Extended IP and network management documentation*. Univention GmbH. URL: https://docs.software-univention.de/ext-networks/5.0/en/.

[15] Jelmer R. Vernooij, John H. Terpsta and Gerald (Jerry) Carter. *The Official Samba 3.2.x HOWTO and Reference Guide*, chapter 20 - Hosting a Microsoft Distributed File System Tree, pages 381–384. Samba Project, May 2009. URL: https://www.samba.org/samba/docs/Samba3-HOWTO.pdf.

[16] *Univention OX Connector app documentation*. Univention GmbH, 2023. URL: https://docs.software-univention.de/ox-connector-app/latest/.

INDEX

A

--append
    udm command line option, 84
auth/faillog, 127
auth/faillog/limit, 127
auth/faillog/lock_global, 127
auth/faillog/root, 128
auth/faillog/unlock_time, 127
auth/sshd/user/root, 169

B

backup/clean/max_age, 40
bonding
    network, 154
bridge
    network, 153

C

commit
    ucr command line option, 159
connector/ad/ldap/binddn, 182, 184, 188
connector/ad/ldap/bindpw, 183, 188
connector/ad/ldap/ssl, 185
connector/ad/mapping/allowsub-
        tree/.*/ad, 193
connector/ad/mapping/allowsub-
        tree/.*/ucs, 193
connector/ad/mapping/group/language,
        195
connector/ad/mapping/ignoresub-
        tree/.*, 193
connector/ad/mapping/{type}/allow-
        filter, 193
connector/ad/mapping/{type}/ignore-
        filter, 194
connector/ad/mapping/{type}/ig-
        norelist, 194
connector/ad/poll/sleep, 183
connector/ad/retryrejected, 183
cron
    syntax, 167
cups/cups-pdf/anonymous, 255
cups/cups-pdf/cleanup/enabled, 255
cups/cups-pdf/cleanup/keep, 255, 292
cups/cups-pdf/directory, 255
cups/errorpolicy, 250
cups/include/local, 250
cups/server, 165

D

-dcaccount
    univention-join command line op-
        tion, 30
-dcname
    univention-join command line op-
        tion, 30
-dcpwd
    univention-join command line op-
        tion, 30
directory/manager/block-
        list/cleanup/cron, 132
directory/manager/blocklist/enabled,
        131
directory/manager/mail-address/uniqueness,
        262
directory/manager/templates/al-
        phanum/whitelist, 130
directory/manager/user_group/unique-
        ness, 110, 137
directory/manager/web/modules/com-
        puters/computer/wizard/dis-
        abled, 143
directory/manager/web/mod-
        ules/groups/group/checks/cir-
        cular_dependency, 139
directory/manager/web/mod-
        ules/users/user/proper-
        ties/mailPrimaryAddress/re-
        quired, 106
directory/manager/web/mod-
        ules/users/user/wizard/dis-
        abled, 106
directory/reports/logo, 89
--dn
    udm command line option, 83
DNS record
    _pkgdb._tcp, 103
dns/allow/transfer, 214
dns/backend, 214
dns/debug/level, 213
dns/dlz/debug/level, 213
dns/forwarder1, 153

dns/forwarder2, 159
dns/forwarder3, 153
dump
    ucr command line option, 158

E

environment variable
    auth/faillog, 127, 291
    auth/faillog/limit, 127, 291
    auth/faillog/lock_global, 127, 291
    auth/faillog/root, 128, 291
    auth/faillog/unlock_time, 127, 291
    auth/sshd/user/root, 169, 291
    backup/clean/max_age, 40, 291
    connector/ad/ldap/binddn, 182, 184, 188, 291
    connector/ad/ldap/bindpw, 183, 188, 291
    connector/ad/ldap/ssl, 185, 291
    connector/ad/mapping/allowsubtree/.*/ad, 190, 193
    connector/ad/mapping/allowsubtree/.*/ucs, 190, 193
    connector/ad/mapping/group/language, 195, 291
    connector/ad/mapping/ignoresubtree/.*, 192, 193
    connector/ad/mapping/{type}/allowfilter, 191, 193
    connector/ad/mapping/{type}/ignorefilter, 192, 194
    connector/ad/mapping/{type}/ignorelist, 192, 194
    connector/ad/poll/sleep, 183, 292
    connector/ad/retryrejected, 183, 292
    cups/cups-pdf/anonymous, 255, 292
    cups/cups-pdf/cleanup/enabled, 255, 292
    cups/cups-pdf/cleanup/keep, 255, 292
    cups/cups-pdf/directory, 255, 292
    cups/errorpolicy, 250, 292
    cups/include/local, 250, 292
    cups/server, 165, 292
    directory/manager/blocklist/cleanup/cron, 132, 292
    directory/manager/blocklist/enabled, 131, 292
    directory/manager/mail-address/uniqueness, 262
    directory/manager/templates/alphanum/whitelist, 130, 292
    directory/manager/user_group/uniqueness, 110, 137, 292
    directory/manager/web/modules/computers/computer/wizard/disabled, 143, 292
    directory/manager/web/modules/groups/group/checks/circular_dependency, 139, 293
    directory/manager/web/modules/users/user/properties/mailPrimaryAddress/required, 106
    directory/manager/web/modules/users/user/wizard/disabled, 106, 293
    directory/reports/cleanup/age, 88
    directory/reports/cleanup/cron, 88
    directory/reports/logo, 89, 293
    dns/allow/transfer, 214, 293
    dns/backend, 214, 293
    dns/debug/level, 213, 293
    dns/dlz/debug/level, 213, 293
    dns/forwarder1, 153, 293
    dns/forwarder2, 159, 293
    dns/forwarder3, 153, 293
    fetchmail/autostart, 269, 293
    freeradius/auth/helper/ntlm/debug, 235, 293
    freeradius/conf/allow-mac-address-authenticati 230, 293
    freeradius/conf/mac-addr-regexp, 232, 293
    freeradius/vlan-id, 232, 235, 294
    gateway, 152, 294
    google-apps/attributes/anonymize, 207, 294
    google-apps/attributes/mapping/.*, 207, 294
    google-apps/attributes/never, 207, 294
    google-apps/debug/werror, 209, 294
    google-apps/groups/sync, 209, 294
    groups/default/domainadmins, 198, 294
    grub/append, 150, 294
    grub/bootsplash, 150, 294
    grub/gfxmode, 150, 294
    grub/timeout, 150, 294
    grub/xenhopt, 150, 294
    http_proxy, 155
    https_proxy, 155
    interfaces/ethX/address, 152, 294
    interfaces/ethX/ipv6/acceptRA, 152, 295
    interfaces/ethX/ipv6/address, 152, 295
    interfaces/ethX/ipv6/prefix, 152, 295
    interfaces/ethX/netmask, 152, 295
    interfaces/ethX/type, 152, 295
    interfaces/ethX_Y/setting, 152, 295
    ipv6/gateway, 152, 295
    kerberos/adminserver, 44, 295
    kerberos/kdc, 44, 295
    kerberos/realm, 44, 295

kernel/blacklist, 149, 295
kernel/modules, 149, 295
ldap/acl/nestedgroups, 39, 296
ldap/acl/read/anonymous, 39, 295
ldap/acl/read/ips, 39, 295
ldap/acl/user/passwordreset/accesslist/groups/dn, 39, 296
ldap/acl/user/passwordreset/attributes, 39, 296
ldap/acl/user/passwordreset/protected/gid, 39, 296
ldap/acl/user/passwordreset/protected/uid, 39, 296
ldap/database/internal/acl/blocklists/groups/read, 133, 295
ldap/database/internal/acl/blocklists/groups/write, 133, 295
ldap/idletimeout, 38, 296
ldap/logging/exclude1, 37, 296
ldap/logging/excludeN, 37, 296
ldap/logging/id-prefix, 38, 296
ldap/master, 50, 296
ldap/overlay/lastbind, 130, 296
ldap/overlay/lastbind/precision, 130, 296
ldap/overlay/memberof/memberof, 141, 296
ldap/policy/cron, 160, 296
ldap/ppolicy, 128, 296, 297
ldap/ppolicy/enabled, 128, 297
ldap/pw-bcrypt, 44, 297
ldap/server/addition, 40, 165, 297
ldap/server/name, 40, 165, 297
listener/debug/level, 42, 297
listener/shares/rename, 239, 297
local/repository, 99, 297
log/rotate/weeks, 165, 297
logrotate/compress, 165, 297
logrotate/listener-modules/compress, 166
logrotate/listener-modules/create, 165
logrotate/listener-modules/missingok, 166
logrotate/listener-modules/notifempty, 166
logrotate/listener-modules/rotate, 165
logrotate/listener-modules/rotate/count, 165
logrotate/rotates, 165, 297
machine/password/length, 143, 297
mail/antispam/bodysizelimit, 267, 297
mail/antispam/learndaily, 267, 297
mail/antispam/requiredhits, 267, 297
mail/antivir, 267, 297
mail/antivir/spam, 267, 297
mail/archivefolder, 271, 298
mail/dovecot/auth/cache_negative_ttl, 263, 298
mail/dovecot/auth/cache_ttl, 263, 298
mail/dovecot/folder/ham, 267, 298
mail/dovecot/folder/Spam, 267, 298
mail/dovecot/imap, 261, 298
mail/dovecot/limits, 275, 298
mail/dovecot/limits/default_client_limit, 275
mail/dovecot/location/separate_index, 274, 298
mail/dovecot/mailbox/delete, 264, 273, 298
mail/dovecot/mailbox/rename, 264, 273, 298
mail/dovecot/pop3, 261, 298
mail/dovecot/process/dotlock_use_excl, 274, 299
mail/dovecot/process/lock_method, 274, 298
mail/dovecot/process/mail_fsync, 274, 299
mail/dovecot/process/mail_nfs_index, 274, 298
mail/dovecot/process/mail_nfs_storage, 274, 298
mail/dovecot/process/mmap_disable, 274, 299
mail/dovecot/quota/warning/subject, 266, 299
mail/dovecot/quota/warning/text, 266, 299
mail/dovecot/quota/warning/text/80, 266
mail/dovecot/quota/warning/text/95, 267
mail/hosteddomains, 262, 299
mail/messagesizelimit, 271, 299
mail/postfix/mastercf/options/smtp/smtpd_sasl_auth_enable, 271, 299
mail/postfix/policy/listfilter, 264, 299
mail/postfix/postscreen/, 272, 299
mail/postfix/postscreen/enabled, 271, 299
mail/postfix/smtpd/restrictions/recipient, 268, 299
mail/postfix/softbounce, 271, 299
mail/postfix/tls/client/level, 270, 300
mail/relayauth, 270, 300
mail/relayhost, 269, 270, 300
monitoring/dns/lookup-domain, 283
nameserver1, 153, 300
nameserver2, 300
nameserver3, 153, 300
notifier/debug/level, 42, 300

nscd/debug/level, 168, 300
nscd/group/maxdbsize, 168, 300
nscd/group/positive_time_to_live, 168, 300
nscd/hosts/maxdbsize, 168, 300
nscd/hosts/positive_time_to_live, 168, 300
nscd/hosts/size, 168
nscd/passwd/maxdbsize, 168, 300
nscd/passwd/positive_time_to_live, 168, 300
nscd/passwd/size, 168
nscd/threads, 168, 300
nss/group/cachefile/check_member, 139, 300
nss/group/cachefile/invalidate_interval, 139, 301
nss/group/cachefile/invalidate_on_changes, 139, 301
nssldap/bindpolicy, 40, 301
ntp/signed, 169, 301
office365/adconnection/wizard, 206, 301
office365/attributes/anonymize, 204, 205, 301
office365/attributes/mapping/.*, 205, 301
office365/attributes/never, 205, 301
office365/attributes/static/.*, 204, 301
office365/attributes/sync, 204, 205, 301
office365/attributes/usageLocation, 204, 301
office365/debug/werror, 206, 301
office365/defaultalias, 206, 301
office365/groups/sync, 205, 206, 301
password/hashing/bcrypt, 44, 301
password/hashing/bcrypt/cost_factor, 44, 302
password/hashing/bcrypt/prefix, 44, 302
password/hashing/method, 44, 302
password/quality/credit/digits, 118, 302
password/quality/credit/lower, 118, 302
password/quality/credit/other, 118, 302
password/quality/credit/upper, 118, 302
password/quality/forbidden/chars, 118, 302
password/quality/length/min, 118, 302
password/quality/mspolicy, 118, 302
password/quality/required/chars, 118, 302
pkgdb/scan, 103, 302
portal/auth-mode, 6163, 302
portal/default-dn, 66, 303
portal/reload-tabs-on-logout, 59
proxy/http, 155, 303
proxy/https, 155, 303
proxy/no_proxy, 155, 303
quota/logfile, 247, 303
quota/userdefault, 247, 303
radius/mac/whitelisting, 230, 303
radius/use-service-specific-password, 230, 303
repository/mirror/server, 99, 303
repository/online/component/.*/unmaintained, 98, 303
repository/online/server, 99, 303
samba/enable-msdfs, 245, 303
samba/max/protocol, 172, 303
samba/spoolss/architecture, 256, 303
samba4/sysvol/sync/cron, 173, 304
saml/idp/authsource, 46, 304
saml/idp/entityID/supplement/[identifier], 48, 304
saml/idp/entityID/supplement/secondIDP, 48
saml/idp/negotiate/filter-subnets, 46, 304
saml/idp/selfservice/account-verification/error-descr, 124, 304
saml/idp/selfservice/account-verification/error-title, 124, 304
saml/idp/selfservice/check_email_verification, 124, 304
security/packetfilter/disabled, 226, 304
self-service/backend-server, 119, 122, 124, 304
self-service/ldap_attributes, 120
self-service/udm_attributes, 120
self-service/udm_attributes/read-only, 120
server/password/change, 143, 304
server/password/interval, 143, 304
server/role, 50, 304
squid/allowfrom, 228, 304
squid/auth/allowed_groups, 229, 304
squid/basicauth, 228, 305
squid/cache, 227, 305
squid/httpport, 228, 305
squid/krb5auth, 229, 305
squid/ntlmauth, 228, 305
squid/ntlmauth/keepalive, 229, 305
squid/webports, 228, 305
sshd/permitroot, 168, 305
sshd/port, 169, 305
sshd/xforwarding, 169, 305

ssl/validity/host, 43, 305
ssl/validity/root, 43, 305
ssl/validity/warning, 43, 305
system/stats, 166, 305
system/stats/cron, 166, 305
timeserver, 169, 306
timeserver2, 169, 306
timeserver3, 169, 306
ucr/check/type, 158, 306
ucs/web/theme, 56, 57, 306
umc/cookie-banner/cookie, 68, 69
umc/cookie-banner/domains, 68, 69
umc/cookie-banner/show, 68, 69
umc/cookie-banner/text, 68, 69
umc/cookie-banner/title, 68, 69
umc/http/processes, 64
umc/http/session/timeout, 57, 306
umc/oidc/issuer, 63, 306
umc/oidc/rp/server, 64, 306
umc/self-service/account-deregistration/email/sender_address, 126
umc/self-service/account-deregistration/email/server, 126
umc/self-service/account-deregistration/email/text_file, 126
umc/self-service/account-deregistration/enabled, 126, 306
umc/self-service/account-registration/backend/enabled, 122
umc/self-service/account-registration/udm_attributes, 122
umc/self-service/account-registration/udm_attributes/required, 122
umc/self-service/account-registration/usercontainer, 122
umc/self-service/account-registration/usertemplate, 122
umc/self-service/account-verification/backend/enabled, 124, 306
umc/self-service/account-verification/email/sender_address, 122
umc/self-service/account-verification/email/server, 122
umc/self-service/account-verification/email/text_file, 122
umc/self-service/account-verification/email/token_length, 124
umc/self-service/account-verification/email/webserver_address, 122
umc/self-service/allow-authenticated-use, 120
umc/self-service/passwordreset/backend/enabled, 119
umc/self-service/profiledata/enabled, 120
umc/self-service/protect-account/backend/enabled, 119
umc/self-service/service-specific-passwords/backend/enabled, 119, 230
umc/web/oidc/enabled, 62, 63, 306
umc/web/sso/enabled, 62, 306
users/default/administrator, 198, 306
Errata updates
UCS 4.4 erratum 536, 37
UCS 4.4 erratum 887, 44
UCS 5.0 erratum 974, 131
UCS 5.0 erratum 1011, 232, 293
UCS 5.0 erratum 1060, 205
UCS 5.0 erratum 1118, 61

F

fetchmail/autostart, 269
freeradius/auth/helper/ntlm/debug, 235
freeradius/conf/allow-mac-address-authentication, 230
freeradius/conf/mac-addr-regexp, 232
freeradius/vlan-id, 232, 235

G

gateway, 152
get
ucr command line option, 158
google-apps/attributes/anonymize, 207, 294
google-apps/attributes/mapping/.*, 207, 294
google-apps/attributes/never, 207
google-apps/debug/werror, 209
google-apps/groups/sync, 209
groups/default/domainadmins, 198
grub/append, 150
grub/bootsplash, 150
grub/gfxmode, 150
grub/timeout, 150
grub/xenhopt, 150

H

hostname, 21
allowed characters, 21
Create new UCS domain, 22
Join existing Active Directory domain, 23
Join existing UCS domain, 24
length, 21
naming convention, 21
http_proxy, 155
https_proxy, 155

I

--ignore-exists
udm command line option, 84
interfaces/ethX/address, 152
interfaces/ethX/ipv6/acceptRA, 152
interfaces/ethX/ipv6/address, 152
interfaces/ethX/ipv6/prefix, 152
interfaces/ethX/netmask, 152

interfaces/ethX/type, 152
interfaces/ethX_Y/setting, 152
ipv6/gateway, 152

K

kerberos/adminserver, 44
kerberos/kdc, 44
kerberos/realm, 44
kernel/blacklist, 149
kernel/modules, 149
Knowledge Base
    KB 32, 173
    KB 37, 43
    KB 6439, 141
    KB 6682, 51
    KB 6701, 57
    KB 14404, 130

L

ldap/acl/nestedgroups, 39
ldap/acl/read/anonymous, 39
ldap/acl/read/ips, 39
ldap/acl/user/passwordreset/accesslist/groups/dn, 39
ldap/acl/user/passwordreset/attributes, 39
ldap/acl/user/passwordreset/protected/gid, 39
ldap/acl/user/passwordreset/protected/uid, 39
ldap/database/internal/acl/blocklists/groups/read, 133
ldap/database/internal/acl/blocklists/groups/write, 133
ldap/idletimeout, 38
ldap/logging/exclude1, 37, 296
ldap/logging/excludeN, 37
ldap/logging/id-prefix, 38
ldap/master, 50
ldap/overlay/lastbind, 130
ldap/overlay/lastbind/precision, 130
ldap/overlay/memberof/memberof, 141
ldap/policy/cron, 160
ldap/ppolicy, 128, 297
ldap/ppolicy/enabled, 128, 297
ldap/pw-bcrypt, 44
ldap/server/addition, 40, 165
ldap/server/name, 40, 165
listener/debug/level, 42
listener/shares/rename, 239
local/repository, 99
log/rotate/weeks, 165
logrotate/compress, 165
logrotate/rotates, 165

M

machine/password/length, 143
mail/antispam/bodysizelimit, 267
mail/antispam/learndaily, 267
mail/antispam/requiredhits, 267
mail/antivir, 267
mail/antivir/spam, 267
mail/archivefolder, 271
mail/dovecot/auth/cache_negative_ttl, 263
mail/dovecot/auth/cache_ttl, 263
mail/dovecot/folder/ham, 267
mail/dovecot/folder/Spam, 267
mail/dovecot/imap, 261
mail/dovecot/limits, 275
mail/dovecot/limits/default_client_limit, 275
mail/dovecot/location/separate_index, 274
mail/dovecot/mailbox/delete, 264, 273
mail/dovecot/mailbox/rename, 264, 273
mail/dovecot/pop3, 261
mail/dovecot/process/dotlock_use_excl, 274
mail/dovecot/process/lock_method, 274
mail/dovecot/process/mail_fsync, 274
mail/dovecot/process/mail_nfs_index, 274
mail/dovecot/process/mail_nfs_storage, 274
mail/dovecot/process/mmap_disable, 274
mail/dovecot/quota/warning/subject, 266
mail/dovecot/quota/warning/text, 266
mail/dovecot/quota/warning/text/80, 266
mail/dovecot/quota/warning/text/95, 267
mail/hosteddomains, 262
mail/messagesizelimit, 271
mail/postfix/mastercf/options/smtp/smtpd_sasl_auth_enable, 271
mail/postfix/policy/listfilter, 264
mail/postfix/postscreen/, 272
mail/postfix/postscreen/enabled, 271
mail/postfix/smtpd/restrictions/recipient, 268
mail/postfix/softbounce, 271
mail/postfix/tls/client/level, 270
mail/relayauth, 270
mail/relayhost, 269, 270
monitoring/dns/lookup-domain, 283

N

nameserver1, 153
nameserver3, 153
network
    802.1q, 154
    bonding, 154
    bridge, 153

etherchannel, 154
link aggregation, 154
switch, 153
teaming, 154
trunking, 154
vlan, 154
notifier/debug/level, 42
nscd/debug/level, 168
nscd/group/maxdbsize, 168
nscd/group/positive_time_to_live, 168
nscd/hosts/maxdbsize, 168
nscd/hosts/positive_time_to_live, 168
nscd/hosts/size, 168
nscd/passwd/maxdbsize, 168
nscd/passwd/positive_time_to_live, 168
nscd/passwd/size, 168
nscd/threads, 168
nss/group/cachefile/check_member, 139
nss/group/cachefile/invalidate_interval, 139
nss/group/cachefile/invalidate_on_changes, 139
nssldap/bindpolicy, 40
ntp/signed, 169

O

office365/adconnection/wizard, 206
office365/attributes/anonymize, 204, 205, 301
office365/attributes/mapping/.*, 205
office365/attributes/never, 205
office365/attributes/static/.*, 204
office365/attributes/sync, 204, 205, 301
office365/attributes/usageLocation, 204
office365/debug/werror, 206
office365/defaultalias, 206
office365/groups/sync, 205, 206
--option
udm command line option, 83

P

password/hashing/bcrypt, 44
password/hashing/bcrypt/cost_factor, 44
password/hashing/bcrypt/prefix, 44
password/hashing/method, 44
password/quality/credit/digits, 118
password/quality/credit/lower, 118
password/quality/credit/other, 118
password/quality/credit/upper, 118
password/quality/forbidden/chars, 118
password/quality/length/min, 118
password/quality/mspolicy, 118
password/quality/required/chars, 118
pkgdb/scan, 103
--policy-reference
udm command line option, 83
portal/auth-mode, 6163
portal/default-dn, 66
portal/reload-tabs-on-logout, 59
--position
udm command line option, 83
proxy/http, 155
proxy/https, 155
proxy/no_proxy, 155

Q

quota/logfile, 247
quota/userdefault, 247

R

radius/mac/whitelisting, 230
radius/use-service-specific-password, 230
--remove
udm command line option, 84
repository/mirror/server, 99
repository/online/component/.*/unmaintained, 98
repository/online/server, 99
RFC
RFC 1001, 225
RFC 1002, 225
RFC 3580, 234

S

samba/enable-msdfs, 245
samba/max/protocol, 172
samba/spoolss/architecture, 256
samba4/sysvol/sync/cron, 173
saml/idp/authsource, 46
saml/idp/entityID/supplement/[identifier], 48
saml/idp/entityID/supplement/secondIDP, 48
saml/idp/negotiate/filter-subnets, 46
saml/idp/selfservice/account-verification/error-descr, 124
saml/idp/selfservice/account-verification/error-title, 124
saml/idp/selfservice/check_email_verification, 124
search
ucr command line option, 158
security/packetfilter/disabled, 226
self-service/backend-server, 119, 122, 124
self-service/ldap_attributes, 120
self-service/udm_attributes, 120
server/password/change, 143
server/password/interval, 143
server/role, 50
set

ucr command line option, 158
--set
udm command line option, 83
shell
ucr command line option, 159
squid/allowfrom, 228
squid/auth/allowed_groups, 229
squid/basicauth, 228
squid/cache, 227
squid/httpport, 228
squid/krb5auth, 229
squid/ntlmauth, 228
squid/ntlmauth/keepalive, 229
squid/webports, 228
sshd/permitroot, 168
sshd/port, 169
sshd/xforwarding, 169
ssl/validity/host, 43
ssl/validity/root, 43
ssl/validity/warning, 43
--superordinate
udm command line option, 83
system/stats, 166
system/stats/cron, 166

T

timeserver, 169
timeserver2, 169
timeserver3, 169

U

ucr command line option
commit, 159
dump, 158
get, 158
search, 158
set, 158
shell, 159
unset, 159
ucr/check/type, 158
ucs/web/theme, 56, 57
udm command line option
--append, 84
--dn, 83
--ignore-exists, 84
--option, 83
--policy-reference, 83
--position, 83
--remove, 84
--set, 83
--superordinate, 83
umc/cookie-banner/cookie, 68
umc/cookie-banner/domains, 68
umc/cookie-banner/show, 68
umc/cookie-banner/text, 68
umc/cookie-banner/title, 68
umc/http/processes, 64
umc/http/session/timeout, 57
umc/oidc/issuer, 63
umc/oidc/rp/server, 64
umc/self-service/account-deregistration/enabled, 126
umc/self-service/account-registration/udm_attribu 122
umc/self-service/account-verification/backend/ena 124
umc/self-service/service-specific-passwords/backe 230
umc/web/oidc/enabled, 62, 63
umc/web/sso/enabled, 62
Univention Help
Univention Help 19514, 51
Univention Help 21833, 27
univention-join command line option
-dcaccount, 30
-dcname, 30
-dcpwd, 30
-verbose, 30
unset
ucr command line option, 159
users/default/administrator, 198

V

-verbose
univention-join command line option, 30
vlan
network, 154