OPEN-XCHANGE™ Whitepaper

keep in touch

OX
OPEN-XCHANGE
SERVER 5.0
• Outlook-Echtzeitanbindung
• Integration einer WebDAV-Schnittstelle
• Dedizierters Lösch-/Besitzrecht
• Flexible Frameplatzierung
• Instant Messaging- Komponente
• Implementierung „Öffentlicher Ordner"
• Anlage von Sichtvorlagen
• Erweiterung der Terminserienfunktion
• Teamansicht

Open-Xchange™ Server in Distributed Environments

v1.01

The information contained in this book was compiled with the utmost care. Nevertheless, erroneous statements cannot be excluded altogether. Open-Xchange Inc., the authors and the translators are not liable for possible errors and their consequences.

The names of software and hardware used in this book may be registered trademarks; they are used without guarantee of free usability. Open-Xchange Inc. generally follows the spelling conventions of the manufacturers. The reproduction of brand names, trade names, logos etc. in this book (even without special marking) does not justify the assumption that such names can be considered free (for the purposes of trademark and brand name regulations).

Please direct any recommendations or comments to
documentation@open-xchange.com

Author: Stephan Martin

Editors: Robert Colombara, David Cuthbert

Layout: Robert Colombara

Contents

1. Introduction

Open-Xchange Server 5 is an Open Source-based, full featured Messaging and Collaboration solution that runs on Linux.

The architecture of Open-Xchange Server 5 is completely based on open standards and open protocols and makes use of well known open source services, which are included in the common Linux enterprise distributions, as back-ends.

This whitepaper describes how an Open-Xchange Server can be set up on top of the mentioned Linux services to maximize collaboration capabilities, even across network boundaries and multiple office locations.

Although the Open-Xchange server is able to run in multiple locations, the administration of a system with multiple networks and data replication is a complex task. Therefore the system's administrator should have solid experience with the administration of Linux servers and network infrastructures. Solid knowledge of TCP/IP, Directory Services, DNS and mail routing is required to understand the underlying concepts and cannot be covered in this paper.

2. Architecture Overview, Concepts

This chapter describes the system requirements for a multi location setup and the basic steps to take to meet these requirements.

2.1. Requirements

Efficient usage of a collaboration system across multiple office locations brings some requirements:

- The complete functionality of Open-Xchange Server should be available over each of the multiple locations including methods for conflict handling and collision detection e.g. during the creation of appointments
- The network load on the WAN connections between the different locations needs to be minimized to avoid unnecessary waiting times for the user
- The effect of broken WAN connections between the different locations needs to be minimized so that maximum usability is possible in every location even when the WAN connection is broken
- The consistency of the data in all locations needs to be guaranteed

2.2. Open-Xchange Architecture

The Open-Xchange Server 5 consists of the Open-Xchange Application server, which contains the application logic and which does the main processing work.

All data is stored by back-end services each of which is specially designed to store a particular type of data.

There are four principal types of data storage:

- Directory Service (LDAP)
- Database (SQL)
- Email (IMAP/SMTP)
- File System

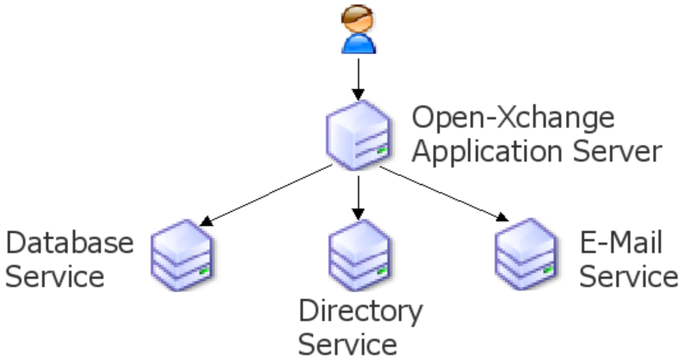

Figure 1: Subsystems inside of the Open-Xchange Server

This architecture permits use of the Open-Xchange Server in multiple locations and delegates the replication of the data to the underlying back-end services, which store the data.

2.3. Multiple Location Concept

The setup for multiple locations with Open-Xchange Server is a hub - spoke model that uses a central location to control several external locations.

The central location runs a read-write instance of the directory server and of the database. All other locations run replicated slaves which are to be accessed read-only.

2.3.1. Normal Operation

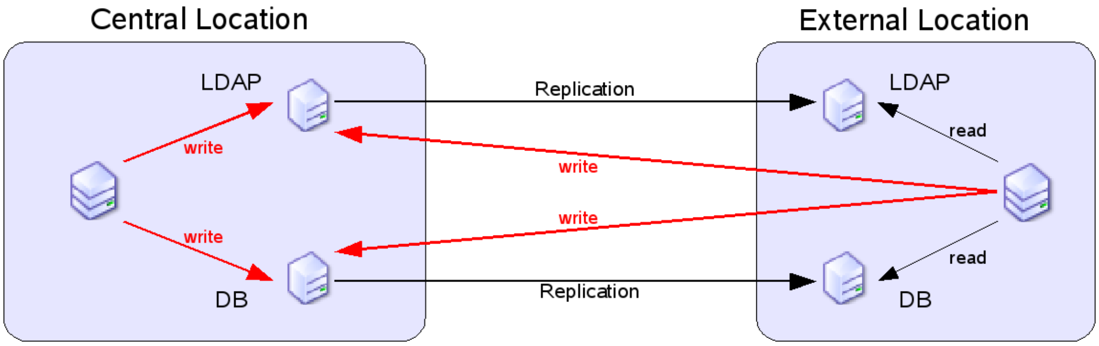

Figure 2: Replication of LDAP Entries during normal Operation

This concept delivers the following functionality during normal operation:

- Every user is able to collaborate with every other user without any boundaries (create group appointments, read shared folders, work with public folders, ...)
- Collision handling and Free-Busy is completely supported between all locations due to the central master database instance
- Consistency of the user data is assured due to the control provided by the central, master LDAP instance
- Read access in the external locations is fast and cheap through access to the local replicated slave LDAP and database
- Email is stored locally, relative to the user’s location; it is not necessary to transfer large emails or attachments over the WAN, when the user accesses his email
- Complete configuration and mail routing information is stored in one LDAP directory with the same layout for all locations. Every location is able to send email directly to the target location without the need of a central relay server

2.3.2. Failure Operation

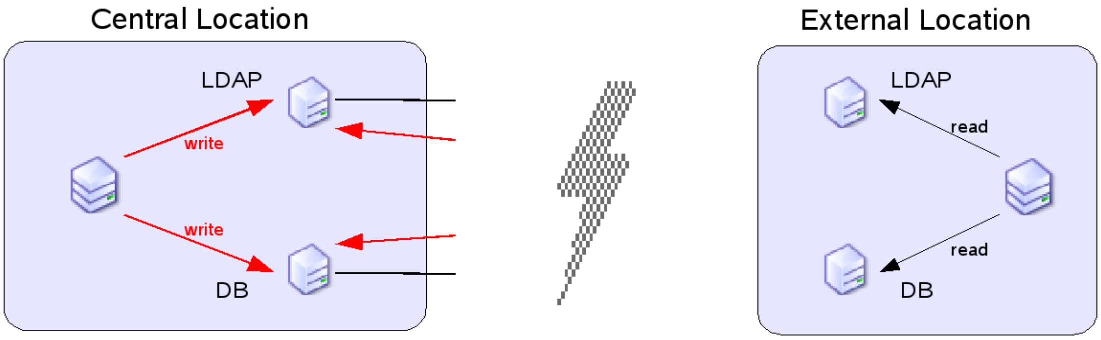

Figure 3: Operation with local Entities disconnected

This concept delivers the following functionality during a failure, caused by network outages, between an external location and the central location:

- Users in still connected locations are able to do their work as usual without any interference
- Users in the external, disconnected locations can still read all their local data (appointments, tasks, contacts, ...)
- Users in disconnected locations cannot write any data though. This is not only due to technical restrictions because the link is down. It would not be possible even if a full multi-master mechanism was used for the directory or the database.
  If the link is down it is not possible to do sufficient conflict resolution checking while writing data. If the written data was successfully queued by a replication subsystem, the user would see a positive result while the connection was down. When the connection was restored and the data is submitted to the servers the user could easily miss warning or error notifications. This means that it would not be possible to guarantee conflict avoidance.
- System administration will not be possible from the disconnected location for the same reasons mentioned above.

3. Service Configuration

This chapter gives some hints and examples on how to configure the Open-Xchange Services and the underlying back-end services to run in a multi-location setup

Basically only a few simple steps need to be done:

1. Install all machines with the same parameters regarding domains, etc. Ensure, that every location has a sub domain name under your main domain like `location1.test.tux` and `location2.test.tux`
2. Configure the Open-Xchange services in the external locations to use the central server for write accesses to the database and the LDAP directory
3. Configure the LDAP server and the database server in the external location to work as slave replicates from the central server.

Note: This concept relies on persistent replication. Any data changed in the external location will be written from the slave server directly to the master server. The slave still depends on instant local availability of the data. This means that a scheduled replication mechanism synchronizing the servers in time slices is not sufficient. Real-time replication is necessary.

3.1. Open-Xchange Services

The Open-Xchange services are enhanced in such a way that they can be configured to *write* to a different LDAP server or database server than the one that is used for *read* access. This is completely transparent to the user.

For LDAP access, this is achieved through a second configuration file called `ldapwrite.conf`. It has the same syntax as the primary file `ldap.conf`. If this second file exists, the application server will separate the read from the write access.

For database connections, the database access definitions in the file `server.conf` need to be duplicated. The duplicates of the variables will contain the string `_WRITE_`, e.g. `NAS_WRITE_CON_CLASS_NAME`. If these duplicated variables exist the application server will separate the read from the write access.

3.2. OpenLDAP Replication

This whitepaper also gives some short hints about the ways to replicate OpenLDAP trees to different servers. There are two primary alternatives, both of which have their advantages and their disadvantages:

- slurpd
  Using the older mechanism, the master LDAP process slapd writes all changes to a log file. A separate server, slurpd runs for each replicated slave server. It reads the log file and applies all changes recorded in the log file to the slave server.
  The disadvantage of this system is that there is no initial sync. This means, every time, the process of replications starts, the contents of the database of master and slave need to be exactly the same. This sounds trivial, but has at least one serous side effect: in every case of a failure and a restart of the replication it is necessary to transfer a complete dump of the master database to the slave for initialization. The other main disadvantage of this mechanism is that the state of both systems is checked using the objects' timestamps. This means that the replication process will move out of sync if the time on master and slave differs.
  The main advantage of this mechanism is that it has been used for quite a long time and it can be considered very stable when set up correctly.
- syncrepl
  SyncRepl is a new replication mechanism based on the standard LDAP replication protocol. In this case, the slave does a search on the master to locate all objects that should be replicated. The state shared between both servers is maintained by a cookie that is stored after each successful replication step. With this mechanism the slave will automatically be initialized if it is started in replication mode without any data. Since the initialization is done automatically, it makes recovery after a failure much easier to handle.

  The main disadvantage of this mechanism is that it is quite new in OpenLDAP and that it may eventually be considered to be less stable than the slurpd mechanism.

3.3. PostgreSQL Replication

Replication of the PostgreSQL database can be achieved with the Open Source replicator slony.

As slony is not available as package for either Red Hat or SUSE Linux, it may be necessary to build your own package for replication. Open-Xchange software partners who specialize in PostgreSQL database solutions can provide slony packages with maintenance and support.

4. Administration

This chapter will give some hints regarding the setup and administration of the system.

The most important fact for administration is that the Open-Xchange administration front-end can be used on every server in the setup. However, to avoid the conflicts mentioned above, it is necessary to direct the write accesses from all of the administration front-ends to the central master server back-ends.

This leads to the fact that administration is possible only as long as the communications link is up and the location is online.

4.1. Domain Setup

Mail routing works in such a way that all email is stored locally, with respect to the user, on the server at his location.

For this mail routing to work it is important to create one sub domain for each location that is only used for internal routing purposes. From an external point of view, all locations will receive and send email with the standard email addresses belonging to the users. The sub domains will not be exposed to any of the recipients; they are only used for internal routing purposes.

In the following example, we have a company with the main domain `test.tux`. Only this domain is exposed in the external communication.

This company has two locations. The central location has the internal sub domain `central.test.tux` and the external location has the internal sub domain `external.test.tux`.

1. These sub domains are created through the Open-Xchange administration front-end, just like normal domains.
2. A transport is created in the Open-Xchange administration front-end for all of the sub domains, which defines the IP address of the server that is responsible for the relevant sub domain.

To allow this mail routing to work transparently for the users and without the need to configure every server in a different way, each server needs to have exactly the same information about every user and every domain. This is easily achieved through the LDAP server, which contains all necessary information and which is replicated to every location.

4.2. User Configuration

The user configuration is straight forward. Each user needs three attributes to define his location and his email servers, a third attribute is dedicated to the multi location setup.

All the email addresses that are assigned to the users are based on the standard top domain and not on the sub domains.

These attributes are set through the Open-Xchange administration front-end.

- `imapServer`
  This attribute defines which IMAP server should be accessed from the Open-Xchange web front-ends. This attribute exists in every Open-Xchange installation and defaults to `localhost`.
- `smtpServer`
  This attribute defines which SMTP server should be accessed from the Open-Xchange web front-ends. This attribute exists in every Open-Xchange installation and defaults to `localhost`.
- `colocRouteAddress`
  This attribute is dedicated to the multi-location setup and contains two parameters in the form of an email address: the user's mailbox name on the local server where the user is physically located and the internal sub domain of the user's location.
  Example: `user@external.test.tux`. Postfix will resolve every email address or email alias for the user on every server in the setup to the location defined by this attribute. This means, in fact, that every server is able to forward the email to the server `external.test.tux`, where the user's mailbox is physically located. This server accepts the email locally and delivers it into the mailbox with the name 'user'.