



# OTOBO Installation Guide

Release 11.0

Rother OSS GmbH

mai 13, 2024



<b>1</b>	<b>Introdução</b>	<b>3</b>
1.1	Sobre esse manual . . . . .	3
<b>2</b>	<b>Requisitos de Hardware e Software</b>	<b>5</b>
2.1	Requisitos de Hardware . . . . .	7
2.2	Requisitos de software . . . . .	7
<b>3</b>	<b>Instalação do OTOBO</b>	<b>9</b>
3.1	Preparação: Desative SELinux quando estiver instalado e ativado . . . . .	9
3.2	Passo 1: Desempacotar e Instalar o OTOBO . . . . .	10
3.3	Passo 2: Instale Programas Adicionais e Módulos Perl . . . . .	10
3.4	Passo 3: Crie o usuário OTOBO . . . . .	11
3.5	Passo 4: Ative o arquivo de configuração padrão . . . . .	11
3.6	Passo 5: Configure seu servidor Web Apache . . . . .	11
3.6.1	Configurar Apache sem suporte SSL . . . . .	12
3.6.2	Configure o Apache <b>com</b> suporte SSL . . . . .	13
3.7	Passo 6: Configurando Permissões de Arquivos . . . . .	13
3.8	Passo 7: Configurando o Banco de Dados . . . . .	13
3.9	Passo 8: Configurar Elasticsearch . . . . .	15
3.9.1	Exemplo de instalação do Elasticsearch baseado no Ubuntu 18.04 LTS . . . . .	15
3.9.2	Instalação do ElasticSearch em outra distribuição Linux . . . . .	15
3.9.3	Instalação do Módulo do ElasticSearch . . . . .	15
3.9.4	Configuração do Elasticsearch . . . . .	15
3.10	Step 9: Basic System Configuration . . . . .	16
3.11	Step 10: First Login . . . . .	16
3.12	Step 11: Start the OTOBO Daemon . . . . .	16
3.13	Step 12: Cron jobs for the OTOBO user . . . . .	16
3.14	Step 13: Setup Bash Auto-Completion (optional) . . . . .	17
3.15	Step 14: Further Information . . . . .	17
<b>4</b>	<b>Instalando usando Docker e Docker Compose</b>	<b>19</b>
4.1	Requerimentos . . . . .	19
4.2	Instalação . . . . .	20
4.2.1	1. Clone the otobo-docker repo . . . . .	20
4.2.2	2. Create an initial .env file . . . . .	20
4.2.3	3. Configure the password for the database admin user . . . . .	21
4.2.4	4. Set up a volume with SSL configuration for the nginx webproxy (optional) . . . . .	21

4.2.5	5. Start the Docker containers with Docker Compose . . . . .	22
4.2.6	6. Install and start OTOBO . . . . .	22
4.3	Informações técnicas adicionais . . . . .	22
4.3.1	Lista de contêineres Docker . . . . .	22
4.3.2	Visão geral dos volumes do Docker . . . . .	23
4.3.3	Variáveis de ambiente do Docker . . . . .	23
4.4	Tópicos avançados . . . . .	24
4.4.1	Configuração personalizada do nginx webproxy . . . . .	24
4.4.2	Single Sign On Using the Kerberos Support in Nginx . . . . .	26
4.4.3	Escolhendo portas não padrão . . . . .	26
4.4.4	Skip startup of specific services . . . . .	26
4.4.5	Prepare offline installation . . . . .	27
4.4.6	Customizing OTOBO Docker Compose . . . . .	27
4.4.7	Personalizando a imagem do OTOBO Docker . . . . .	27
4.4.8	Construindo imagens locais . . . . .	29
4.4.9	Instalação automática . . . . .	29
4.4.10	Lista de comandos úteis . . . . .	30
4.5	Recursos . . . . .	30
<b>5</b>	<b>Migration from OTRS 6 or OTRS 7 / ((OTRS)) Community Edition to OTOBO version 10.1</b>	<b>33</b>
5.1	Overview over the Supported Migration Szenarios . . . . .	33
5.2	Requerimentos de migração . . . . .	34
5.3	Passo 1: Instalação do novo Sistema OTOBO . . . . .	35
5.4	Step 2: Deactivate <code>SecureMode</code> on OTOBO . . . . .	36
5.5	Step 3: Stop the OTOBO Daemon . . . . .	36
5.6	Optional Step: Mount <code>/opt/otrs</code> for Convenient Access . . . . .	36
5.7	Optional Step: Install <code>sshpas</code> and <code>rsync</code> when <code>/opt/otrs</code> Should be Copied via <code>ssh</code> . . . . .	37
5.8	Step 4: Preparing the OTRS / ((OTRS)) Community Edition system . . . . .	37
5.8.1	Interromper todos os serviços relevantes e o Daemon OTRS . . . . .	37
5.8.2	Clear the Caches and the Operational Data . . . . .	38
5.9	Optional Step for Docker: make required data available inside container . . . . .	38
5.9.1	Copie <code>/opt/otrs</code> dentro do volume <code>otobo_opt_otobo</code> . . . . .	38
5.10	Step 5: Perform the Migration! . . . . .	39
5.11	Step 6: After Successful Migration! . . . . .	40
5.12	Known Migration Problems . . . . .	40
5.12.1	1. Login after migration not possible . . . . .	40
5.12.2	2. Final page of the migration has a strange layout due to missing CSS files . . . . .	40
5.12.3	3. Migration stops due to MySQL errors . . . . .	40
5.12.4	4. Errors in Step 5 when migrating to PostgreSQL . . . . .	41
5.12.5	5. Problems with the Deployment the Merged System Configuration . . . . .	41
5.13	Etapa 7: Tarefas de migração manual e alterações . . . . .	41
5.13.1	1. Password policy rules . . . . .	41
5.13.2	2. Under Docker: Manually migrate cron jobs . . . . .	42
5.14	Special topics . . . . .	42
5.14.1	Migration from Oracle to Oracle . . . . .	42
5.14.2	Optional Step: Streamlined migration of the database (only for experts and spezial szenarios) . . . . .	43
<b>6</b>	<b>Atualizando</b>	<b>47</b>
6.1	Etapa 1: interromper todos os serviços relevantes e o OTOBO Daemon . . . . .	47
6.2	Etapa 2: Arquivos de backup e banco de dados . . . . .	48
6.2.1	Exemplo de instalação padrão com Ubuntu e MySQL . . . . .	48
6.3	Etapa 3: instalar a nova versão . . . . .	48
6.3.1	Restaurar arquivos de configuração antigos . . . . .	48

6.3.2	Restaurar dados do artigo	49
6.3.3	Restaurar estatísticas padrão já instaladas	49
6.3.4	Definir permissões de arquivo	49
6.3.5	Check Apache configuration files	49
6.4	Step 4: Check for new needed perl modules	49
6.5	Step 5: Update Installed Packages and reconfigure config	50
6.6	Step 6: Only for minor or major release upgrades (for example to upgrade from 10.0 to 10.1)	50
6.7	Step 7: Start your Services	50
<b>7</b>	<b>Atualizando uma instalação baseada em Docker do OTOBO</b>	<b>51</b>
7.1	Atualizar os arquivos Docker Compose	51
7.2	Verificando o arquivo Docker Compose .env	52
7.3	Busque as novas imagens do Docker	52
7.4	Atualize o OTOBO	52
<b>8</b>	<b>Backup e Restauração</b>	<b>55</b>
8.1	Backup	55
8.2	Restaurar	56
8.3	Considerações para executar OTOBO no Docker	56
<b>9</b>	<b>Backup e Restauração usando Docker</b>	<b>59</b>
9.1	Considerações para executar OTOBO no Docker	59
<b>10</b>	<b>Kerberos Single Sign On in OTOBO Docker installation</b>	<b>61</b>
10.1	Generate Active Directory User	61
10.2	Generate Active Directory Keytab file	61
10.3	Create a new volume for your custom nginx configuration	63
10.4	Create new OTOBO .env file	63
10.5	Start OTOBO	64
10.6	Tell OTOBO to use the Kerberos-Authentication	64
10.7	Configure Browser to understand Kerberos SSO	64
10.8	Debugging and Problems	65
10.8.1	Kerberos debugging	65
<b>11</b>	<b>Adapt customer interface with corporate identity</b>	<b>67</b>
11.1	Change colors in Customer Area	67
11.2	Change Logos and Pictures	67
11.2.1	Change Customer Login Pictures and Text	68
11.2.2	Change Customer Dashboard tiles and options	69
<b>12</b>	<b>Installing Perl Modules from CPAN</b>	<b>71</b>
12.1	Docker-based installations	71
<b>13</b>	<b>Ajuste de desempenho</b>	<b>73</b>
13.1	Módulo de Índice do Ticket	73
13.2	Índice de pesquisa de tickets	73
13.3	Pesquisa de documentos	75
13.3.1	Tamanho Heap	75
13.3.2	Alocação de disco	76
13.4	Armazenamento de artigos	77
13.5	Arquivando Tickets	77
13.6	Armazenamento em cache	78
13.6.1	Instale um servidor de cache Redis	78
13.6.2	Cache do RamDisk	79

13.7 Clustering . . . . .	79
<b>14 Histórico da documentação</b>	<b>81</b>



Este trabalho é protegido por direitos autorais OTRS AG (<https://otrs.com>), Zimmersmühlenweg 11, 61440 Oberursel, Germany.

Copyright © for modifications and amendments 2019-2022 ROTHER OSS GmbH (<https://otobo.de>), Oberwalting 31, 94339 Leiblfing, Germany

Termos e Condições OTRS: É concedida permissão para copiar, distribuir e/ou modificar este documento sob os termos da GNU Free Documentation License, Versão 1.3 ou qualquer versão posterior publicada pela Free Software Foundation; sem secções Invariantes, sem textos de capa e sem textos de contracapa. Uma cópia da licença pode ser encontrada no site GNU <<https://www.gnu.org/licenses/fdl-1.3.txt>> \_\_.

Termos e Condições Rother OSS: É concedida permissão para copiar, distribuir e / ou modificar este documento sob os termos da GNU Free Documentation License, Versão 1.3 ou qualquer versão posterior publicada pela Free Software Foundation; sem seções invariantes, sem textos de capa e sem textos de contracapa. Uma cópia da licença está incluída na seção “CÓPIA”.

Publicado por: Rother OSS GmbH, (<https://otobo.de>), Oberwalting 31, 94339 Leiblfing, Germany.

Autores: OTRS AG (versão original), Rother OSS GmbH (<https://otobo.de>).





OTOBO é um sistema de solicitação de tickets de código aberto com muitos recursos para gerenciar chamadas telefônicas e e-mails de clientes. É distribuído sob a GNU General Public License (GPL) e é testado em várias plataformas Linux.

### 1.1 Sobre esse manual

Este manual é destinado ao uso de administradores de sistema. Os capítulos descrevem a instalação e atualização do software OTOBO.

Não há interface gráfica com o usuário para instalação e atualização. Os administradores de sistema precisam seguir as etapas descritas nos capítulos seguintes.

Todos os comandos do console se parecem com `username> command-to-execute`. Nome de usuário indica a conta de usuário do sistema operacional, que precisa ser usada para executar o comando. Se um comando iniciar com `root>`, você deverá executar o comando como um usuário com permissões de root. Se um comando iniciar com `otobo>`, você deverá executar o comando como o usuário criado para o OTOBO.

**Aviso:** Não selecione “username>” quando você copia o comando e o cola no shell. Caso contrário, você receberá um erro.

Assumimos que o OTOBO será instalado no diretório `/opt/otobo`. Se você deseja instalar o OTOBO em um local diferente, você deve alterar o caminho nos comandos ou criar um link simbólico para este diretório.

```
root> ln -s /path/to/otobo /opt/otobo
```



---

## Requisitos de Hardware e Software

---

The OTOBO web application can be installed on Linux and other Unix derivates, e.g. OpenBSD or FreeBSD. Running OTOBO on Microsoft Windows is not supported.

The web application uses a relational database as backend. So, to run OTOBO, you'll need to run at least a web server and a database server. The web server and the database server may be installed either on the same or on different hosts.

Alternatively, OTOBO can also run under Docker. When running under Docker, the web and the database server are already included in the setup. Support for deployment with Kubernetes is under development.

The OTOBO web application requires Perl along with additional Perl modules from CPAN. The modules can be installed either with a Perl package manager or via the package manager of your operating system (rpm, yast, apt-get). There is a console command for checking the module dependencies:

```
otobo> /opt/otobo/bin/otobo.CheckModules.pl --inst
```

If some packages are missing, you can get an install command for your operating system by running the script with the `--list` option.

```
otobo> /opt/otobo/bin/otobo.CheckModules.pl --list | more
```

Os comandos listados devem ser executados com privilégios de root.

A saída do script de verificação do módulo mostra os pacotes instalados e os números da versão. Os módulos ausentes estão marcados com um comentário.

```
Required packages:
  o Archive::Tar.....ok (v2.32)
  o Archive::Zip.....ok (v1.67)
  o Const::Fast.....ok (v0.014)
  o Date::Format.....ok (v2.24)
  o DateTime.....ok (v1.51)
    o DateTime::TimeZone.....ok (v2.38)
  o Convert::BinHex.....ok (v1.125)
```

- o DBI.....ok (v1.643)
- o Digest::SHA.....ok (v6.02)
- o File::chmod.....ok (v0.42)
- o List::AllUtils.....ok (v0.15)
- o LWP::UserAgent.....ok (v6.26)
- o Moo.....ok (v2.003006)
- o namespace::autoclean.....ok (v0.29)
- o Net::DNS.....ok (v1.22)
- o Net::SMTP::SSL.....ok (v1.04)
- o Path::Class.....ok (v0.37)
- o Sub::Exporter.....ok (v0.987)
- o Template::Toolkit.....ok (undef)
- o Template::Stash::XS.....ok (undef)
- o Text::CSV.....ok (v1.95)
- o Text::Trim.....ok (v1.04)
- o Time::HiRes.....ok (v1.9760)
- o Try::Tiny.....ok (v0.30)
- o URI.....ok (v1.71)
- o XML::LibXML.....ok (v2.0207)
- o YAML::XS.....ok (v0.81)
- o Unicode::Collate.....ok (v1.27)
- o CGI::PSGI.....ok (v0.15)
- o DBIx::Connector.....ok (v0.56)
- o Path::Class.....ok (v0.37)
- o Plack.....ok (v1.0047)
- o Plack::Middleware::ForceEnv.....ok (v0.02)
- o Plack::Middleware::Header.....ok (v0.04)
- o Plack::Middleware::Refresh.....ok (undef)
- o Plack::Middleware::ReverseProxy.....ok (v0.16)
- o Plack::Middleware::Rewrite.....ok (v2.101)
- o SOAP::Transport::HTTP::Plack.....ok (v0.03)

Recommended features for setups using apache:

- o ModPerl::Util.....ok (v2.000011)

Database support (installing one is required):

- o DBD::mysql.....ok (v4.050)

Various features for additional functionality:

- o Encode::HanExtra.....ok (v0.23)
- o Net::LDAP.....ok (v0.66)
- o Crypt::Eksblowfish::Bcrypt.....ok (v0.009)
- o XML::LibXSLT.....ok (v1.99)
- o XML::Parser.....ok (v2.46)

Features enabling communication with a mail-server:

- o Net::SMTP.....ok (v3.11)
- o Mail::IMAPClient.....ok (v3.42)
- o Authen::SASL.....ok (v2.16)
- o Authen::NTLM.....ok (v1.09)
- o IO::Socket::SSL.....ok (v2.067)

Optional features which can increase performance:

- o JSON::XS.....ok (v4.02)
- o Text::CSV\_XS.....ok (v1.41)

Required packages if you want to use PSGI/Plack (experimental and advanced):

- o Gazelle.....ok (v0.49)

```
o Linux::Inotify2.....ok (v2.2)
o Plack::App::File.....ok (undef)
```

## 2.1 Requisitos de Hardware

Os requisitos de hardware dependem muito do uso do OTOBO. O OTOBO pode ser usado para processar alguns tickets por mês ou para processar centenas de tickets por dia. O requisito de armazenamento também depende do número de tickets e do tamanho dos anexos.

Recomendamos o uso de uma máquina para fins de teste com **pelo menos**:

- small CPU
- 4 GB RAM
- 10 GB storage

Recomendamos o uso de uma máquina para fins de produção com **pelo menos**:

- CPU 3 GHz Xeon ou comparável
- 8 GB RAM (16 GB recomendado)
- 40 GB armazenamento

---

**Nota:** Os requisitos de hardware dependem do uso do OTOBO. Entre em contato com seu consultor OTOBO antes de implantar qualquer hardware.

---

## 2.2 Requisitos de software

### Perl

- Perl 5.24.0 ou superior
- Perl packages listed by `/opt/otobo/bin/otobo.CheckModules.pl --list console command`

### Web Server

- Apache HTTP Server Version 2.4

### Bases de dados

- MySQL 5.6 ou superior
- MariaDB
- PostgreSQL 9.2 ou superior
- Oracle 10g ou superior

### Opcional

- Elasticsearch 7.x (função de pesquisa rápida para pré-visualizações ao vivo)
- Redis (cache rápido)
- nginx or any other web server that can be used as a reverse proxy (SSL support and load distribution)

## Navegadores

- Apple Safari
- Google Chrome
- Microsoft Internet Explorer 11
- Microsoft Edge
- Mozilla Firefox
- Qualquer outro navegador moderno com suporte a JavaScript

---

## Instalação do OTOBO

---

Este capítulo descreve a instalação e a configuração básica da estrutura central do OTOBO.

Siga as etapas detalhadas neste capítulo para instalar o OTOBO no seu servidor. Você pode usar sua interface da web para efetuar login e administrar o sistema.

---

**Nota:** As of OTOBO version 10.0.7, we recommend Docker and Docker Compose for the OTOBO installation. By using the provided Docker images, all recommended dependencies (such as Elasticsearch, Redis Cache, etc.) are installed and configured automatically. Updates are thus greatly simplified and the performance has been improved. You can find the instructions for Docker-based installation at <https://doc.otobo.org/manual/installation/10.1/en/content/installation-docker.html>.

---

### 3.1 Preparação: Desative SELinux quando estiver instalado e ativado

---

**Nota:** Se seu sistema usa o SELinux, você deve desativá-lo, caso contrário o OTOBO não funcionará corretamente.

---

Tente os comandos “`sestatus`” e `getenforce` quando você não tiver certeza se o SELinux está instalado e ativado no seu sistema.

O comando `sestatus` retorna o estado do SELinux e a política do SELinux que está sendo usada. Estado do SELinux: `enabled` é retornado quando o SELinux está ativado. Modo atual: `enforcing` é retornado quando o SELinux está sendo executado no modo `enforcing`. Política do arquivo de configuração: `target` é retornado quando a política de destino do SELinux é usada.

Veja como desativar o SELinux para RHEL/CentOS/Fedora.

1. Configurar `SELINUX=disabled` no arquivo `/etc/selinux/config` :

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#     targeted - Targeted processes are protected,
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2. Reinicie seu sistema. Após reiniciar, confirme que o comando “getenforce” retorne Disabled:

```
root> getenforce
Disabled
```

## 3.2 Passo 1: Desempacotar e Instalar o OTOBO

Baixe a última versão do otobo em <https://ftp.otobo.org/pub/otobo/>. Descompacte o arquivo fonte (por exemplo, usando “tar”) dentro do diretório /opt/otobo-install:

```
root> mkdir /opt/otobo-install && mkdir /opt/otobo # Create a
↳temporary install directory
root> cd /opt/otobo-install # Change into
↳the update directory
root> wget https://ftp.otobo.org/pub/otobo/otobo-latest-10.1.tar.gz # Download he
↳latest OTOBO 10 release
root> tar -xzf otobo-latest-10.1.tar.gz # Unzip OTOBO
root> cp -r otobo-10.x.x/* /opt/otobo # Copy the
↳new otobo directory to /opt/otobo
```

## 3.3 Passo 2: Instale Programas Adicionais e Módulos Perl

Use o seguinte script para obter uma visão geral de todos os módulos CPAN instalados e requeridos e outras dependências externas.

**Nota:** On Debian systems you may need to manually install some perl packages:

```
apt-get install -y libarchive-zip-perl libtimedate-perl libdatettime-perl libconvert-
↳binhex-perl libcgi-psgi-perl libdbi-perl libdbix-connector-perl libfile-chmod-perl
↳liblist-allutils-perl libmoo-perl libnamespace-autoclean-perl libnet-dns-perl
↳libnet-smtp-ssl-perl libpath-class-perl libsub-exporter-perl libtemplate-perl
↳libtext-trim-perl libtry-tiny-perl libxml-libxml-perl libyaml-libyaml-perl libdbd-
↳mysql-perl libapache2-mod-perl2 libmail-imapclient-perl libauthen-sasl-perl
↳libauthen-ntlm-perl libjson-xs-perl libtext-csv-xs-perl libpath-class-perl libplack-
↳perl libplack-middleware-header-perl libplack-middleware-reverseproxy-perl
↳libencode-hanextra-perl libio-socket-ssl-perl libnet-ldap-perl libcrypt-eksblowfish-
↳perl libxml-libxslt-perl libxml-parser-perl libconst-fast-perl
```



```
root> perl /opt/otobo/bin/otobo.CheckModules.pl -list
Checking for Perl Modules:
  o Archive::Tar.....ok (v1.90)
  o Archive::Zip.....ok (v1.37)
  o Crypt::Eksblowfish::Bcrypt.....ok (v0.009)
...
```

**Nota:** Observe que o OTOBO requer uma instalação funcionando do Perl com todos os módulos core, como o módulo `version`. Esses módulos não são explicitamente verificados pelo script. Pode ser necessário instalar um pacote `perl-core` em alguns sistemas como o RHEL que não instalam os pacotes principais do Perl por padrão.

Para instalar os pacotes obrigatórios e opcionais, você pode usar o CPAN ou o gerenciador de pacotes da sua distribuição Linux.

Execute este comando para obter um comando de instalação para instalar as dependências ausentes:

```
root> /opt/otobo/bin/otobo.CheckModules.pl --inst
```

**Nota:** Existem vários módulos opcionais ou alternativos que podem ser instalados, principalmente para versões mais personalizadas do OTOBO. Chamar `CheckModules.pl` sem nenhum argumento listará sua funcionalidade completa.

### 3.4 Passo 3: Crie o usuário OTOBO

Crie um usuário dedicado para OTOBO dentro de seu próprio grupo:

```
root> useradd -r -U -d /opt/otobo -c 'OTOBO user' otobo -s /bin/bash
```

Adicione o usuário no grupo do seu servidor web (se o seu servidor web não estiver rodando com o usuário `otobo`):

```
root> usermod -G www-data otobo
(SUSE=www, Red Hat/CentOS/Fedora=apache, Debian/Ubuntu=www-data)
```

### 3.5 Passo 4: Ative o arquivo de configuração padrão

Há um arquivo de configuração OTOBO incluído no `$OTOBO_HOME/Kernel/Config.pm.dist`. Você deve ativá-lo copiando o arquivo sem a extensão do arquivo `.dist`.

```
root> cp /opt/otobo/Kernel/Config.pm.dist /opt/otobo/Kernel/Config.pm
```

### 3.6 Passo 5: Configure seu servidor Web Apache

Em primeiro lugar, você deve instalar o servidor Web Apache2 e o `mod_perl`; você normalmente faz isso no gerenciador de pacotes do seu sistema. Abaixo, você encontrará os comandos necessários

para configurar o Apache nas distribuições Linux mais populares.

```
# RHEL / CentOS:
root> yum install httpd mod_perl

# SuSE:
root> zypper install apache2-mod_perl

# Debian/Ubuntu:
root> apt-get install apache2 libapache2-mod-perl2
```

A critical setting of the Apache web server is the choice of the multi-processing module. For running OTOBO, the recommended choice is the module **mpm\_prefork**. Like other Apache modules the multi-processing module can be managed with the tools `a2dismod` and `a2enmod`.

```
root> # check which MPM is active
root> apache2ctl -M | grep mpm_
```

Tudo está bem quando `mpm_prefork` já foi habilitado.

Desative `mpm_event` quando estiver ativo no momento.

```
root> a2dismod mpm_event
```

Desative `mpm_worker` caso o MPM esteja ativo.

```
root> a2dismod mpm_worker
```

Finalmente ative `mpm_prefork`.

```
root> a2enmod mpm_prefork
```

O OTOBO requer que mais alguns módulos Apache estejam ativos para uma operação ideal. Novamente, na maioria das plataformas, você pode ter certeza de que estão ativas por meio da ferramenta `a2enmod`.

```
root> a2enmod perl
root> a2enmod deflate
root> a2enmod filter
root> a2enmod headers
```

---

**Nota:** Em algumas plataformas, nem todos os módulos Apache existem e um erro será exibido durante a instalação. Não se preocupe e termine a instalação, na maioria dos casos o módulo não será necessário.

---

A maioria das instalações do Apache tem um diretório `conf.d` incluído. Em sistemas Linux, você geralmente pode encontrar este diretório em `/etc/apache` ou `/etc/apache2`.

### 3.6.1 Configurar Apache sem suporte SSL

Copie o arquivo de modelo `/opt/otobo/scripts/apache2-httpd.include.conf` para o diretório `sites-available` do apache. Na maioria dos casos, nenhuma edição adicional do modelo é necessária. Em seguida, ative a nova configuração.

```
# Debian/Ubuntu:
root> cp /opt/otobo/scripts/apache2-httpd.include.conf /etc/apache2/sites-available/
↪ zzz_otobo.conf
root> a2ensite zzz_otobo.conf
root> systemctl restart apache2
```

### 3.6.2 Configure o Apache com suporte SSL

Copie os arquivos de modelo `/opt/otobo/scripts/apache2-httpd-vhost-80.include.conf` e `/opt/otobo/scripts/apache2-httpd-vhost-443.include.conf` para o diretório `sites-available`.

```
# Debian/Ubuntu:
root> cp /opt/otobo/scripts/apache2-httpd-vhost-80.include.conf /etc/apache2/sites-
↪ available/zzz_otobo-80.conf
root> cp /opt/otobo/scripts/apache2-httpd-vhost-443.include.conf /etc/apache2/sites-
↪ available/zzz_otobo-443.conf
```

Edite os arquivos e adicione as informações necessárias, como o caminho de armazenamento do certificado SSL. Depois disso, ative a configuração do OTOBO Apache:

```
root> a2ensite zzz_otobo-80.conf
root> a2ensite zzz_otobo-443.conf
```

Agora você pode reiniciar o servidor da web para carregar as novas definições de configuração. Na maioria dos sistemas, você pode usar o seguinte comando para fazer isso:

```
root> systemctl restart apache2
```

## 3.7 Passo 6: Configurando Permissões de Arquivos

Por favor, execute o seguinte comando para definir as permissões de arquivo e diretório para o OTOBO. Ele tentará detectar as configurações corretas de usuário e grupo necessárias para sua configuração.

```
root> /opt/otobo/bin/otobo.SetPermissions.pl
```

## 3.8 Passo 7: Configurando o Banco de Dados

Primeiro de tudo, você deve instalar o pacote de banco de dados. É recomendável usar o pacote MySQL ou MariaDB, que será entregue com o sistema Linux, mas é possível usar o PostgreSQL ou Oracle também.

Você normalmente faria isso no gerenciador de pacotes do sistema. Abaixo, você encontrará os comandos necessários para configurar o MySQL nas distribuições Linux mais populares.

```
# RHEL / CentOS:
root> yum install mysql-server

# SuSE:
root> zypper install mysql-community-server
```

```
# Debian/Ubuntu:
root> apt-get install mysql-server
```

Depois de instalar o servidor MySQL, você precisa configurá-lo.

Na versão 5.7 ou superior do MySQL, um novo módulo de autenticação está ativo e não é possível usar o instalador da web OTOBO para criação de banco de dados. Por favor, faça login no console do mysql e defina um módulo de autenticação e senha diferentes para o usuário `root`, se este for o caso:

```
root> mysql -u root
root> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY
↵ 'NewRootPassword';
```

Para MariaDB > 10.1, use o seguinte comando:

```
root> mysql -u root
root> update mysql.user set authentication_string=password('NewRootPassword') plugin=
↵ 'mysql_native_password' where user='root';
```

Se este comando não funcionar, tente os seguintes comandos:

```
root> mysql -u root
root> UPDATE mysql.user SET password = PASSWORD('NewRootPassword') WHERE user = 'root
↵';
root> UPDATE mysql.user SET authentication_string = '' WHERE user = 'root';
root> UPDATE mysql.user SET plugin = 'mysql_native_password' WHERE user = 'root';
```

Após a instalação do OTOBO, é possível alterar o módulo de autenticação novamente, se necessário.

**Nota:** As seguintes definições de configuração são requisitos mínimos para as configurações do MySQL. Por favor, adicione as seguintes linhas ao arquivo de configuração do MySQL Server `/etc/my.cnf`, `/etc/mysql/my.cnf` ou `/etc/mysql/mysql.conf.d/mysqld.cnf` abaixo da seção `[mysqld]`:

```
max_allowed_packet = 64M
innodb_log_file_size = 256M
```

Para o MySQL anterior ao MySQL 8.0, o tamanho do cache da consulta também deve ser definido:

```
query_cache_size = 32M
```

Além disso, adicione as seguintes linhas ao arquivo de configuração do servidor MySQL `/etc/my.cnf`, `/etc/mysql/my.cnf` ou `/etc/mysql/mysql.conf.d/mysqldump.cnf` debaixo da seção `[mysqldump]`:

```
max_allowed_packet = 64M
```

Para fins de produção, recomendamos usar a ferramenta `mysqltuner` para encontrar a configuração perfeita. Você pode fazer o download do script no github <https://github.com/major/MySQLTuner-perl> ou instalá-lo nos sistemas Debian ou Ubuntu através do gerenciador de pacotes:

```
root> apt-get install mysqltuner
```

Após a instalação execute o script:

```
root> mysqltuner --user root --pass NewRootPassword
```

## 3.9 Passo 8: Configurar Elasticsearch

A OTOBO recomenda uma instalação ativa do Elasticsearch para pesquisa rápida. A maneira mais fácil é configurar o Elasticsearch no mesmo host que o OTOBO e vinculá-lo à sua porta padrão.

### 3.9.1 Exemplo de instalação do Elasticsearch baseado no Ubuntu 18.04 LTS

Instalação do JDK

```
root> apt update
root> apt install openjdk-8-jdk
```

Instalação do ElasticSearch

```
root> wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key_
↵add -
root> echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee_
↵/etc/apt/sources.list.d/elasticsearch-7.x.list
root> apt update
root> apt -y install elasticsearch
```

### 3.9.2 Instalação do ElasticSearch em outra distribuição Linux

Siga o tutorial de instalação encontrado em <https://www.elastic.co/guide/en/elasticsearch/reference/current/setup.html>.

### 3.9.3 Instalação do Módulo do ElasticSearch

Além disso, o OTOBO exige a instalação de plug-ins no Elasticsearch:

```
root> /usr/share/elasticsearch/bin/elasticsearch-plugin install --batch ingest-
↵attachment
root> /usr/share/elasticsearch/bin/elasticsearch-plugin install --batch analysis-icu
```

### 3.9.4 Configuração do Elasticsearch

O Elasticsearch possui diversas opções e possibilidades de configuração.

Para garantir uma operação livre de erros, você deve ajustar o espaço de heap da jvm para sistemas OTOBO maiores. Por favor, ajuste as configurações no arquivo `/etc/elasticsearch/jvm.options`. Você sempre deve definir o tamanho mínimo e máximo do heap da JVM para o mesmo valor. Por exemplo, para definir o heap como 4 GB, defina:

```
-Xms4g
-Xmx4g
```

Em nossos testes, um valor entre 4 e 10 GB para instalações de tamanho médio provou ser o melhor.

**Nota:** Veja <https://www.elastic.co/guide/en/elasticsearch/reference/current/heap-size.html> para mais informações.

Agora você pode reiniciar o servidor da web para carregar as novas definições de configuração. Na maioria dos sistemas, você pode usar o seguinte comando para fazer isso:

```
root> systemctl restart elasticsearch
```

## 3.10 Step 9: Basic System Configuration

Por favor, use o instalador da web em <http://localhost/otobo/installer.pl> (substitua “localhost” pelo seu nome de host OTOBO) para configurar seu banco de dados e configurações básicas do sistema, como contas de email.

## 3.11 Step 10: First Login

Agora você está pronta para logar no seu sistema em <http://localhost/otobo/index.pl> com o usuário `root@localhost` com a senha que foi gerada (veja acima).

## 3.12 Step 11: Start the OTOBO Daemon

O daemon OTOBO é responsável por manipular qualquer tarefa assíncrona e recorrente no OTOBO. O que havia sido definido anteriormente nos arquivos cron agora é tratado pelo daemon OTOBO, necessário para operar o OTOBO. O daemon também manipula todas as tarefas GenericAgent e deve ser iniciado a partir do usuário OTOBO.

```
otobo> /opt/otobo/bin/otobo.Daemon.pl start
```

## 3.13 Step 12: Cron jobs for the OTOBO user

Existem dois arquivos cron OTOBO padrão em `/opt/otobo/var/cron/*.dist`, e seu objetivo é garantir que o Daemon OTOBO esteja em execução. Eles precisam ser ativados, copiando-os sem a extensão de arquivo “.dist”.

```
root> cd /opt/otobo/var/cron/  
root> for foo in *.dist; do cp $foo `basename $foo .dist`; done  
  
root> cd /opt/otobo/  
root> bin/Cron.sh start
```

Com esta etapa, a configuração básica do sistema está concluída.

### 3.14 Step 13: Setup Bash Auto-Completion (optional)

Todas as operações regulares da linha de comando OTOBO acontecem através da interface do console OTOBO. Isso fornece um preenchimento automático para o shell bash, o que facilita a localização do comando e das opções corretas.

Você pode ativar o auto-completion do bash instalando o pacote `bash-completion`. Ele detectará e carregará o arquivo automaticamente `/opt/otobo/.bash_completion` para o usuário `otobo`.

Após reiniciar seu shell, você pode apenas digitar este comando, seguido de TAB, e ele listará todos os comandos disponíveis:

```
otobo> /opt/otobo/bin/otobo.Console.pl
```

Se você digitar alguns caracteres do nome do comando, o TAB exibirá todos os comandos correspondentes. Após digitar um comando completo, todas as opções e argumentos possíveis serão mostrados pressionando TAB.

**Nota:** Se tiver problemas, você pode executar a seguinte linha como usuário `otobo` e adicioná-la ao seu `~/ .bashrc` para executar os comandos do arquivo.

```
source /opt/otobo/.bash_completion
```

### 3.15 Step 14: Further Information

Recomendamos que você leia o capítulo [OTOBO Ajuste de desempenho](#).





---

## Instalando usando Docker e Docker Compose

---

With the dockerized OTOBO deployment you can get your personal OTOBO instance up and running within minutes. All of OTOBO's dependencies are already included in the provided collection of Docker images.

- Service db: MariaDB is set up as the default database.
- Service elastic: Elasticsearch is set up for the OTOBO power search.
- Service redis: Redis is enabled for fast caching.
- Service web: Gazelle is used as fast Perl webserver.
- Service nginx: Nginx is used as optional reverse proxy for HTTPS support.

We think that this setup is the perfect environment for an OTOBO installation.

### 4.1 Requerimentos

As versões mínimas do software necessário, que foram testadas, estão listadas aqui:

- Docker 19.03.13
- Docker Compose 1.25.0
- Git 2.17.1

---

**Nota:** Para obter as versões mínimas necessárias no Ubuntu 18.04, siga as instruções em <https://www.digitalocean.com/community/tutorials/how-to-install-docker-compose-on-ubuntu-18-04> e <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-18-04>.

---

git, Docker e Docker Compose podem ser instalados com as ferramentas de sistema padrão. Aqui está um exemplo de instalação no Ubuntu 20.04:

```
root> apt-get install git docker docker-compose curl
root> systemctl enable docker
```

Verifique a documentação do Git e do Docker para obter instruções sobre configurações adicionais.

## 4.2 Instalação

As instruções a seguir pressupõem que todos os requisitos sejam atendidos, que você tenha um ambiente Docker funcionando. Presumimos aqui que o usuário **docker\_admin** é usado para interagir com o Docker. O administrador do Docker pode ser o usuário **root** do host do Docker ou um usuário dedicado com as permissões necessárias.

### 4.2.1 1. Clone the otobo-docker repo

The Docker images will eventually be fetched from the repository <https://hub.docker.com>. But there are some setup and command files that need to be cloned from the otobo-docker Github repository. Make sure that you specify the branch that corresponds to the current version of OTOBO. For example, when OTOBO 10.0.15 is the current version then please use the branch `rel-10_0`.

---

**Nota:** A localização do repositório clonado não importa. Para essas instruções, escolhemos `/opt/otobo-docker` como o diretório de trabalho.

---

```
docker_admin> cd /opt
docker_admin> git clone https://github.com/RotherOSS/otobo-docker.git --branch
↪<BRANCH> --single-branch
docker_admin> ls otobo-docker      # just a sanity check, README.md should exist
```

### 4.2.2 2. Create an initial .env file

The Docker Compose configuration file `.env` is your primary interface for managing your installation of OTOBO. This file must first be created and then be adapted by yourself. In order to simplify the task there are several example files that should be used as starting point. Which sample file it the best fit depends on your use case. In most cases the decision is between `.docker_compose_env_http` and `.docker_compose_env_https`, depending on whether TLS must be supported or not. The other files are for more specialised use cases.

**.docker\_compose\_env\_http** O aplicativo da web OTOBO fornece HTTP.

**.docker\_compose\_env\_https** The OTOBO web app provides HTTPS by running Nginx as a reverse proxy webserver.

**.docker\_compose\_env\_https\_custom\_nginx** Like `.docker_compose_env_https` but with support for a custom Nginx configuration.

**.docker\_compose\_env\_https\_kerberos** Like `.docker_compose_env_https` but with sample setup for single sign on. Note that Kerberos support is still **experimental**.

**.docker\_compose\_env\_http\_selenium and .docker\_compose\_env\_https\_selenium** These are used only for development when Selenium testing is activated.

---

**Nota:** Use `ls -a` for listing the hidden sample files.

---

Por padrão, o OTOBO é servido nas portas padrão. Porta 443 para HTTPS e porta 80 para HTTP. Quando o HTTPS é ativado, o aplicativo da web OTOBO ainda é executado com HTTP. O suporte HTTPS é obtido por um proxy reverso adicional, que é implementado como um serviço nginx.

Para os comandos a seguir, presumimos que HTTPS deve ser compatível.

```
docker_admin> cd /opt/otobo-docker
docker_admin> cp -p .docker_compose_env_https .env # or .docker_compose_env_http for
↳ HTTP
```

### 4.2.3 3. Configure the password for the database admin user

Altere o seguinte valor dentro do arquivo `.env`:

```
OTOBO_DB_ROOT_PASSWORD=<your_secret_password>
```

A senha do usuário administrador do banco de dados pode ser escolhida livremente. O usuário administrador do banco de dados é necessário para criar o usuário do banco de dados **otobo** e o esquema do banco de dados **otobo**. O OTOBO realmente usará o usuário de banco de dados dedicado **otobo**.

### 4.2.4 4. Set up a volume with SSL configuration for the nginx webproxy (optional)

Esta etapa pode ser ignorada quando o OTOBO deve estar disponível apenas via HTTP.

O nginx precisa de um certificado e uma chave privada para criptografia SSL.

---

**Nota:** Para teste e desenvolvimento, um certificado auto-assinado pode ser usado. No entanto, para uso produtivo, você deve trabalhar com certificados regulares registrados.

Veja, por exemplo <https://www.digitalocean.com/community/tutorials/how-to-create-a-self-signed-ssl-certificate-for-nginx-in-ubuntu-18-04> sobre como criar certificados auto-assinados.

---



---

**Nota:** Para especificar uma cadeia de CA com um certificado no nginx, é necessário copiar o arquivo de cadeia de CA com o certificado real em um arquivo.

---

O certificado e a chave privada são armazenados em um volume, para que possam ser usados pelo nginx posteriormente. Em qualquer caso, o volume precisa ser gerado manualmente e precisamos copiar o certificado e a chave para o volume:

```
docker_admin> docker volume create otobo_nginx_ssl
docker_admin> otobo_nginx_ssl_mp=$(docker volume inspect --format '{{ .Mountpoint }}'
↳ otobo_nginx_ssl)
docker_admin> echo $otobo_nginx_ssl_mp # just a sanity check
docker_admin> cp /PathToYourSSLCert/ssl-cert.crt /PathToYourSSLCert/ssl-key.key
↳ $otobo_nginx_ssl_mp
```

Os nomes dos arquivos copiados precisam ser definidos em nosso arquivo `.env` recém-criado. Por exemplo.

```
OTOBO_NGINX_SSL_CERTIFICATE=/etc/nginx/ssl/ssl-cert.crt and  
OTOBO_NGINX_SSL_CERTIFICATE_KEY=/etc/nginx/ssl/ssl-key.key
```

Adapte apenas o nome dos arquivos, pois o caminho `/etc/nginx/ssl/` é codificado na imagem do Docker.

### 4.2.5 5. Start the Docker containers with Docker Compose

Now we start the Docker containers using `docker-compose`. Per default the Docker images will be fetched from <https://hub.docker.com/u/rotheross>.

```
docker_admin> docker-compose up --detach
```

Para verificar se os seis serviços necessários (cinco no caso de HTTP apenas) estão realmente em execução, digite:

```
docker_admin> docker-compose ps  
docker_admin> docker volume ls
```

### 4.2.6 6. Install and start OTOBO

Execute o instalador OTOBO em `http://yourIPorFQDN/otobo/installer.pl`.

---

**Nota:** Please configure OTOBO inside the installer with a new MySQL database. As MySQL database root password please use the password you configured in the variable `OTOBO_DB_ROOT_PASSWORD` of your `.env` file. Please leave the value `db` for the MySQL hostname untouched.

---

#### Divirta-se com o OTOBO!

---

**Nota:** To change to the OTOBO directory, inside the running container, to work on command line as usual, you can use the following Docker command: `docker-compose exec web bash`.

---

## 4.3 Informações técnicas adicionais

Esta seção oferece mais informações técnicas sobre o que está acontecendo nos bastidores.

### 4.3.1 Lista de contêineres Docker

**Contêiner `otobo_web_1`** Servidor web OTOBO na porta interna 5000.

**Contêiner `otobo_daemon_1`** Daemon OTOBO. O daemon OTOBO é iniciado e verificado periodicamente.

**Contêiner `otobo_db_1`** Executa o banco de dados MariaDB na porta interna 3306.

**Contêiner `otobo_elastic_1`** Elasticsearch nas portas internas 9200 e 9300.

**Contêiner otobo\_redis\_1** Execute o Redis como serviço de cache.

**Contêiner opcional otobo\_nginx\_1** Execute o nginx como proxy reverso para fornecer suporte HTTPS.

### 4.3.2 Visão geral dos volumes do Docker

Os volumes Docker são criados no host para dados persistentes. Isso permite iniciar e interromper os serviços sem perder dados. Lembre-se de que os contêineres são temporários e apenas os dados nos volumes são permanentes.

**otobo\_opt\_otobo** contains /opt/otobo in the container **web** and **daemon**.

**otobo\_mariadb\_data** contains /var/lib/mysql in the container **db**.

**otobo\_elasticsearch\_data** contains /usr/share/elasticsearch/data in the container **elastic**.

**otobo\_redis\_data** contém dados para o contêiner redis.

**otobo\_nginx\_ssl** contém os arquivos TLS, certificado e chave privada, deve ser inicializado manualmente.

### 4.3.3 Variáveis de ambiente do Docker

In the instructions we did only minimal configuration. But the file .env allows to set more variables. Here is a short list of the most important environment variables. Note that more environment variables are supported by the base images.

#### Configurações MariaDB

**OTOBO\_DB\_ROOT\_PASSWORD** The root password for MariaDB. This setting is required for running the service db.

#### Configurações do Elasticsearch

Elasticsearch precisa de algumas configurações para ambientes de produção. Por favor leia <https://www.elastic.co/guide/en/elasticsearch/reference/7.8/docker.html#docker-prod-prerequisites> para informações detalhadas.

**OTOBO\_Elasticsearch\_ES\_JAVA\_OPTS** Configuração de exemplo: OTOBO\_Elasticsearch\_ES\_JAVA\_OPTS=-Xms512m -Xmx512m Ajuste este valor para env de produção para um valor de até 4g.

**\*\*Configurações do Webserver \*\***

**OTOBO\_WEB\_HTTP\_PORT** Definido caso a porta HTTP deva se desviar da porta 80 padrão. Quando HTTPS está habilitado, a porta HTTP redireciona para HTTPS.

#### Nginx webproxy settings

Essas configurações serão usadas quando HTTPS está habilitado.

**OTOBO\_WEB\_HTTP\_PORT** Definido caso a porta HTTP se desvie da porta 80 padrão. Irá redirecionar para HTTPS.

**OTOBO\_WEB\_HTTPS\_PORT** Definido no caso de a porta HTTPS se desviar da porta 443 padrão.

**OTOBO\_NGINX\_SSL\_CERTIFICATE** Certificado SSL para o webproxy nginx. Exemplo:  
OTOBO\_NGINX\_SSL\_CERTIFICATE=/etc/nginx/ssl/acme.crt

**OTOBO\_NGINX\_SSL\_CERTIFICATE\_KEY** Chave SSL para o webproxy nginx. Exemplo:  
OTOBO\_NGINX\_SSL\_CERTIFICATE\_KEY=/etc/nginx/ssl/acme.key

### Nginx webproxy settings for Kerberos

This settings are used by Nginx when Kerberos is used for single sign on.

**OTOBO\_NGINX\_KERBEROS\_KEYTAB** Kerberos keytab file. The default is /etc/krb5.keytab.

**OTOBO\_NGINX\_KERBEROS\_CONFIG** Kerberos config file. The default is /etc/krb5.conf, usually generated from krb5.conf.template

**OTOBO\_NGINX\_KERBEROS\_SERVICE\_NAME** Kerberos Service Name. It is not clear where this setting is actually used anywhere.

**OTOBO\_NGINX\_KERBEROS\_REALM** Kerberos REALM. Used in /etc/krb5.conf.

**OTOBO\_NGINX\_KERBEROS\_KDC** Kerberos kdc / AD Controller. Used in /etc/krb5.conf.

**OTOBO\_NGINX\_KERBEROS\_ADMIN\_SERVER** Kerberos Admin Server. Used in /etc/krb5.conf.

**OTOBO\_NGINX\_KERBEROS\_DEFAULT\_DOMAIN** Kerberos Default Domain. Used in /etc/krb5.conf.

**NGINX\_ENVSUBST\_TEMPLATE\_DIR** Provide a custom Nginx config template dir. Gives extra flexibility.

### Configurações do Docker Compose

Essas configurações são usadas diretamente pelo Docker Compose.

**COMPOSE\_PROJECT\_NAME** The project name is used as the prefix for the volumes and containers. Per default this prefix is set to otobo, resulting in container names like otobo\_web\_1 and otobo\_db\_1. Change this name when you want to run more then one instance of OTOBO on the same server.

**COMPOSE\_PATH\_SEPARATOR** Separador para o valor de COMPOSE\_FILE

**COMPOSE\_FILE** Use docker-compose/otobo-base.yml como base e adicione os arquivos de extensão desejados. Por exemplo docker-compose/otobo-override-http.yml ou docker-compose/otobo-override-https.yml.

**OTOBO\_IMAGE\_OTOBO, OTOBO\_IMAGE\_OTOBO\_ELASTICSEARCH, OTOBO\_IMAGE\_OTOBO\_NGINX, ...** Used for specifying alternative Docker images. Useful for testing local builds or for using updated versions of the images.

## 4.4 Tópicos avançados

### 4.4.1 Configuração personalizada do nginx webproxy

The container `otobo_nginx_1` provides HTTPS support by running Nginx as a reverse proxy. The Docker image that runs in the container is composed of the official Nginx Docker image, [https://hub.docker.com/\\_/nginx](https://hub.docker.com/_/nginx), along with a OTOBO specific configuration of Nginx.

The default OTOBO specific configuration can be found within the Docker image at `/etc/nginx/template/otobo_nginx.conf.template`. Actually, this is only a template for the final configuration. There is a process, provided by the Nginx base image, that replaces the macros in the template with the corresponding environment variable. This process runs when the container starts up. In the default template file, the following macros are used:

**OTOBO\_NGINX\_SSL\_CERTIFICATE** For configuring SSL.

**OTOBO\_NGINX\_SSL\_CERTIFICATE\_KEY** For configuring SSL.

**OTOBO\_NGINX\_WEB\_HOST** The internally used HTTP host.

**OTOBO\_NGINX\_WEB\_PORT** The internally used HTTP port.

See step 4. for how this configuration possibility was used for setting up the SSL certificate.

**Aviso:** A abordagem a seguir só é compatível com OTOBO 10.0.4 ou posterior.

When the standard macros are not sufficient, then the customisation can go further. This can be achieved by replacing the default config template with a customized version. It is best practice to not simply change the configuration in the running container. Instead we first create a persistent volume that contains the custom config. Then we tell the `otobo_nginx_1` to mount the new volume and to use the customized configuration.

First comes generation of the new volume. In these sample commands, we use the existing template as a starting point.

```
# stop the possibly running containers
docker_admin> cd /opt/otobo-docker
docker_admin> docker-compose down

# create a volume that is initially not connected to otopo_nginx_1
docker_admin> docker volume create otopo_nginx_custom_config

# find out where the new volume is located on the Docker host
docker_admin> otopo_nginx_custom_config_mp=$(docker volume inspect --format '{{ .
↳Mountpoint }}' otopo_nginx_custom_config)
docker_admin> echo $otopo_nginx_custom_config_mp # just a sanity check
docker_admin> ls $otopo_nginx_custom_config_mp # another sanity check

# copy the default config into the new volume
docker_admin> docker create --name tmp-nginx-container rotheross/otobo-nginx-
↳webproxy:latest-10_0 # or latest-10_1, use the appropriate label
docker_admin> docker cp tmp-nginx-container:/etc/nginx/templates/otobo_nginx.conf.
↳template $otopo_nginx_custom_config_mp # might need 'sudo'
docker_admin> ls -l $otopo_nginx_custom_config_mp/otobo_nginx.conf.template # just_
↳checking, might need 'sudo'
docker_admin> docker rm tmp-nginx-container

# adapt the file $otopo_nginx_custom_config_mp/otobo_nginx.conf.template to your needs
docker_admin> vim $otopo_nginx_custom_config_mp/otobo_nginx.conf.template
```

**Aviso:** Sua configuração nginx adaptada geralmente contém a diretiva **listen**, que declara as portas do servidor web. As portas usadas internamente foram alteradas entre OTOBO 10.0.3 e OTOBO 10.0.4. Essa mudança deve ser refletida na configuração nginx adaptada. Portanto, para a versão 10.0.3 ou anterior, escute as portas 80 e 443. Para o OTOBO 10.0.4, escute as portas 8080 e 8443.

After setting up the volume, the adapted configuration must be activated. The new volume is set up in `docker-compose/otobo-nginx-custom-config.yml`. Therefore this file must be added to **COMPOSE\_FILE**. Then Nginx must be directed to use the new config. This is done by setting **NGINX\_ENVSUBST\_TEMPLATE\_DIR** in the environment. In order to achieve this, uncomment or add the following lines in your `.env` file:

```
COMPOSE_FILE=docker-compose/otobo-base.yml:docker-compose/otobo-override-https.
↳yml:docker-compose/otobo-nginx-custom-config.yml
NGINX_ENVSUBST_TEMPLATE_DIR=/etc/nginx/config/template-custom
```

A configuração do Docker Compose alterada pode ser inspecionada com:

```
docker_admin> docker-compose config | more
```

Finalmente, os contêineres podem ser iniciados novamente:

```
docker_admin> docker-compose up --detach
```

Consulte também a seção “Using environment variables in nginx configuration (new in 1.19)” em [https://hub.docker.com/\\_/nginx](https://hub.docker.com/_/nginx).

### 4.4.2 Single Sign On Using the Kerberos Support in Nginx

#### Short Description

For enabling authentication with Kerberos please base you .env file on the sample file .docker\_compose\_env\_https\_kerberos. This activates the special configuration in docker-compose/otobo-override-https-kerberos.yml. This Docker compose configuration file selects a Nginx image that supports Kerberos. It also passes some Kerberos specific settings as environment values to the running Nginx container. These settings are listed above.

As usual, the values for these setting can be specified in the .env file. Most of these setting will be used as replacement values for the template [https://github.com/RotherOSS/otobo/blob/rel-10\\_1/scripts/nginx/kerberos/templates/krb5.conf.template](https://github.com/RotherOSS/otobo/blob/rel-10_1/scripts/nginx/kerberos/templates/krb5.conf.template). The replacement takes place during the startup of the container. In the running container the adapted config will be available in /etc/krb5.conf.

Providing an user specific /etc/krb5.conf file is still possible. This can be done by mounting a volume that overrides /etc/krb5.conf in the container. This can be achieved by setting OTOBO\_NGINX\_KERBEROS\_CONFIG in the .env file and by activating the mount directive in docker-compose/otobo-override-https-kerberos.yml.

/etc/krb5.keytab is always installation specific and must therefore always be mounted from the host system.

#### Kerberos SSO Installation Tutorial

[Kerberos Single Sign On in OTOBO Docker installation](#)

### 4.4.3 Escolhendo portas não padrão

Por padrão, as portas 443 e 80 atendem HTTPS e HTTP, respectivamente. Pode haver casos em que uma ou ambas as portas já são usadas por outros serviços. Nestes casos, as portas padrão podem ser substituídas especificando OTOBO\_WEB\_HTTP\_PORT e OTOBO\_WEB\_HTTPS\_PORT no arquivo .env.

### 4.4.4 Skip startup of specific services

The current Docker compose setup start five, six when HTTPS is activated, services. But there are valid use cases where one or more of these services are not needed. The prime example is when the database should not run as a Docker service, but as an external database. Unfortunately there is no dedicated Docker compose option for skipping specific services. But the option -scale can be abused for this purpose. So for an installation with an external database the following command can be used:

```
docker_admin> docker-compose up --detach --scale db=0
```

Of course the same goal can also be achieved by editing the file docker-compose/otobo-base.yml and removing the relevant service definitions.



#### 4.4.5 Prepare offline installation

Please download the latest version of `otobo-docker` on a system that has internet access and where docker is installed. Then navigate to the following folder `otobo-docker/docker-compose`.

```
cd otobo-docker/docker-compose
```

Now you can run the following command to download all Docker images from a specific file, in my example I use the `otobo-base.yml`.

```
for i in $(cat otobo-base.yml | grep image: | cut -d":" -f3,4 | sed -e "s/-//1" -e "s/\}/  
↪/g"); do docker pull $i; docker save $i -o $(echo $i | sed "s/\}/-/g").docker; done
```

After that, the images (`.docker`) are located in the `docker-compose` folder and can be uploaded to the target system via e.g. [SCP](#).

On the offline target system, go to the folder where the docker images are stored. And enter the following command to import them one by one.

In the following example I import the `mariadb` image:

```
docker load --input mariadb:10.5.docker
```

#### 4.4.6 Customizing OTOBO Docker Compose

Instead of editing the files under `docker-compose/` and risking to overwrite your own options with the next update of the `otobo-docker` folder, it is advisable to create an extra YAML file where the specific services are overwritten with additional options.

A common example would be to make the database container accessible from the outside via port 3306. For this you could create an extra docker compose file that looks like:

```
$ cat custom_db.yml
services:
  db:
    ports:
      - "0.0.0.0:3306:3306"
```

Now we have to tell docker-compose to include our new file. For this you have to add your YAML file to the `COMPOSE_FILE` variable in the `.env` file, for example:

```
COMPOSE_FILE=docker-compose/otobo-base.yml:docker-compose/otobo-override-http.  
↪yml:custom_db.yml
```

Now we can use docker-compose to recreate our container

```
$ docker-compose stop # if otobo is running
$ docker-compose up -d
```

With this procedure you can customize any service or volumes.

#### 4.4.7 Personalizando a imagem do OTOBO Docker

Many customizations can be done in the external volume `otobo_opt_otobo` which corresponds to the directory `/opt/otobo` in the Docker image. This works e.g. for local Perl modules which can be installed into `/opt/otobo/local`. Here is an example that installs the not very useful CPAN module `Acme::123`.

```
$ docker exec -it ${COMPOSE_PROJECT_NAME:=otobo}_web_1 bash
otobo@ce36ff89e637:~$ pwd
/opt/otobo
otobo@ce36ff89e637:~$ cpanm -l local Acme::123
--> Working on Acme::123
Fetching http://www.cpan.org/authors/id/N/NA/NATHANM/Acme-123-0.04.zip ... OK
Configuring Acme-123-0.04 ... OK
Building and testing Acme-123-0.04 ... OK
Successfully installed Acme-123-0.04
1 distribution installed
otobo@ce36ff89e637:~$
```

The nice thing of this approach is that the Docker image itself does not have to be modified.

Installing extra Debian packages is a little bit trickier. One approach is to create a custom Dockerfile and use the OTOBO image as the base image. Another approach is to create a modified image directly from a running container. This can be done with the command `docker commit`, <https://docs.docker.com/engine/reference/commandline/commit/>. A nice writeup of that process is available at <https://phoenixnap.com/kb/how-to-commit-changes-to-docker-image>.

But for the latter approach there are two hurdles to overcome. First, the image `otobo` runs per default as the user `otobo` with the UID 1000. The problem is that the user `otobo` is not allowed to install system packages. Thus, the first part of the solution is to pass the option `-user root` when running the image. However the second hurdle is that the default entrypoint script `/opt/otobo_install/entrypoint.sh` exits immediately when it is called as `root`. The reasoning behind that design decision is that running inadvertently as `root` should be discouraged. So, the second part of the solution is to specify a different entrypoint script that does not care who the caller is. This leaves us with following example commands, where we add fortune cookies to `otobo`:

Pull a tagged OTOBO image, if we don't have it yet, and check whether the image already provides fortune cookies:

```
$ docker run rotheross/otobo:rel-10_0_10 /usr/games/fortune
/opt/otobo_install/entrypoint.sh: line 57: /usr/games/fortune: No such file or d
↪directory
```

Add fortune cookies to a named container running the original OTOBO image. This is done in an interactive session as the user `root`:

```
$ docker run -it --user root --entrypoint /bin/bash --name otobo_orig rotheross/
↪otobo:rel-10_0_10
root@50ac203409eb:/opt/otobo# apt update
root@50ac203409eb:/opt/otobo# apt install fortunes
root@50ac203409eb:/opt/otobo# exit
$ docker ps -a | head
```

Create an image from the stopped container and give it a name. Take into account that the default user and entrypoint script must be restored:

```
$ docker commit -c 'USER otobo' -c 'ENTRYPOINT ["/opt/otobo_install/entrypoint.sh"]'
↪otobo_orig otobo_with_fortune_cookies
```

Finally we can doublecheck:

```
$ docker run otobo_with_fortune_cookies /usr/games/fortune
A platitude is simply a truth repeated till people get tired of hearing it.
-- Stanley Baldwin
```

The modified image can be specified in your .env file and then be used for fun and profit.

#### 4.4.8 Construindo imagens locais

**Nota:** Building Docker images locally is usually only needed during development. Other use cases are when more current base images should be used for an installation or when extra functionality must be added to the images.

The Docker files needed for creating Docker images locally are part of the the git repository <https://github.com/RotherOSS/otobo>:

- otopo.web.dockerfile
- otopo.nginx.dockerfile
- otopo.elasticsearch.dockerfile

The script for the actual creation of the images is bin/docker/build\_docker\_images.sh.

```
docker_admin> cd /opt
docker_admin> git clone https://github.com/RotherOSS/otobo.git
docker_admin> # checkout the wanted branch. e.g. git checkout rel-10_0_11
docker_admin> cd otopo
docker_admin> # modify the docker files if necessary
docker_admin> bin/docker/build_docker_images.sh
docker_admin> docker image ls
```

The locally built Docker images are tagged as local-<OTOBO\_VERSION> using the version set up the file RELEASE.

After building the local images, one can return to the docker-compose directory. The local images are declared by setting OTOBO\_IMAGE\_OTOBO, OTOBO\_IMAGE\_OTOBO\_ELASTICSEARCH, OTOBO\_IMAGE\_OTOBO\_NGINX in .env.

#### 4.4.9 Instalação automática

Instead of going through <http://yourIPorFQDN/otobo/installer.pl>, one can take a short cut. This is useful for running the test suite on a fresh installation.

**Aviso:** docker-compose down -v irá remover todas as configurações e dados anteriores.

```
docker_admin> docker-compose down -v
docker_admin> docker-compose up --detach
docker_admin> docker-compose stop daemon
docker_admin> docker-compose exec web bash\
-c "rm -f Kernel/Config/Files/ZZZAAuto.pm ; bin/docker/quick_setup.pl --db-password_
↳otobo_root"
docker_admin> docker-compose exec web bash\
-c "bin/docker/run_test_suite.sh"
.....
docker_admin> docker-compose start daemon
```

## 4.4.10 Lista de comandos úteis

### Docker

- `docker system prune -a` system clean-up (removes all unused images, containers, volumes, networks)
- `docker version` show version
- `docker build --tag otobo --file=otobo.web.Dockerfile .` build an image
- `docker run --publish 80:5000 otobo` run the new image
- `docker run -it -v opt_otobo:/opt/otobo otobo bash` log into the new image
- `docker run -it -v opt_otobo:/opt/otobo --entrypoint bash otobo` try that in case `entrypoint.sh` is broken
- `docker ps` show running images
- `docker images` show available images
- `docker volume ls` list volumes
- `docker volume inspect otobo_opt_otobo` inspect a volume
- `docker volume inspect --format '{{ .Mountpoint }}' otobo_nginx_ssl` get volume mountpoint
- `docker volume rm tmp_volume` remove a volume
- `docker inspect <container>` inspect a container
- `docker save --output otobo.tar otobo:latest-10_0 && tar -tvf otobo.tar` list files in an image
- `docker exec -it nginx-server nginx -s reload` reload nginx

### Docker Compose

- `docker-compose config` check and show the configuration
- `docker-compose ps` show the running containers
- `docker-compose exec nginx nginx -s reload` reload nginx

## 4.5 Recursos

Finally, here is a highly subjective collection of links.

### General info and tutorials

- [Perl Maven: Getting Started with Perl on Docker](#)
- [Dockerfile best practices](#)
- [Environment](#)

### Tips and hints

- [Newer version of Docker Compose on Ubuntu 18.04 LTS](#)
- [Newer version of Docker on Ubuntu 18.04 LTS](#)
- [Clean up unused images](#)

- Docker Host IP
- Self signed certificate

#### **Troubleshooting**

- Docker cache invalidation
- Using tcpdump
- Inspect failed builds



---

## Migration from OTRS 6 or OTRS 7 / ((OTRS)) Community Edition to OTOBO version 10.1

---

**Aviso:** Please migrate your OTRS to OTOBO version 10.1 first and then upgrade your OTOBO to version 11.

Bem-vindo e obrigado por escolher o OTOBO!

OTRS, ((OTRS)) Community Edition e OTOBO são muito abrangentes e flexíveis em sua aplicação. Portanto, toda migração para o OTOBO requer uma preparação completa e, possivelmente, algum retrabalho também.

Reserve um tempo para a migração e siga estas instruções passo a passo.

If you have any problem or question, please do not despair. Call our support line, write an email, or post your query in the OTOBO Community forum at <https://forum.otobo.org/>. We will find a way to help you!

---

**Nota:** After the migration the data previously available in OTRS will be available in OTOBO 10. We do not modify any data of the OTRS installation during the migration.

---

### 5.1 Overview over the Supported Migration Szenarios

With the OTOBO Migration Interface it is possible to employ the following migration strategies:

1. The general migration strategy.

This is the regular way to perform a migration. Many different different combinations are supported:

**Change server:** Migrate and simultaneously move to a new application server.

**Separate application and web servers:** It's your choice whether you want to run application and database server on the same host or each on a dedicated host. This choice is regardless of the previous setup in OTRS / ((OTRS)) Community Edition.

**Different databases:** Migrate from any of the supported databases to any other supported database.

**Different operating system:** Switch from any supported operating system to any other supported operating system.

**Docker:** Migrate to a Docker-based installation of OTOBO 10.

2. A variant of the general strategy where the database migration is streamlined.

Use the ETL-like migration when the source database mustn't suffer from increased load or when access to the source database is a bottleneck. In the general strategy, the data is row by row first read from the otrs database and then inserted into the OTOBO database. In this variant, the complete otrs database tables are first exported, then transformed, and then imported into the otobo database.

3. Migration from an Oracle based OTRS 6 / OTRS 7 installation to an Oracle based OTOBO installation.

This is a special case that is not supported by the general migration strategy. This means that a variant of the streamlined strategy must be used.

**Aviso:** All strategies work for both Docker-based and native installations. But for Docker-based installations some peculiarities have to be considered. These peculiarities are handled in the optional steps.

---

**Nota:** It is also feasible to clone the OTRS data to the OTOBO database server before the actual migration. This can speed up the general migration strategy.

---

## 5.2 Requerimentos de migração

1. Basic requirement for a migration is that you already have an ((OTRS)) Community Edition or OTRS 6.0.\* / OTRS 7.0.\* running, and that you want to transfer both configuration and data to OTOBO.

**Aviso:** Please consider carefully whether you really need the data and configuration. Experience shows that quite often a new start is the better option. This is because in many cases the previously used installation and configuration was rather suboptimal anyways. It might also make sense to only transfer the ticket data and to change the basic configuration to OTOBO Best Practice. We are happy to advise you, please get in touch at [hello@otobo.de](mailto:hello@otobo.de) or ask your question in the OTOBO Community forum at <https://forum.otobo.org/>.

2. Você precisa de uma instalação OTOBO em execução para iniciar a migração a partir daí!
3. Essa instalação do OTOBO deve conter todos os pacotes OPM instalados no seu OTRS que você também deseja usar no OTOBO.



4. If you are planning to migrate to another server, then the OTOBO webserver must be able to access the location where your ((OTRS)) Community Edition or OTRS 6.0.\* / OTRS 7.0.\* is installed. In most cases, this is the directory /opt/otrs on the server running OTRS. The read access can be effected via SSH or via file system mounts.
5. The otrs database must be accessible from the server running OTOBO. Readonly access must be granted for external hosts. If access is not possible, or when the speed of the migration should be optimised, then a dump of the database is sufficient.

---

**Nota:** Se o acesso SSH e ao banco de dados entre ambos servidores não for possível, migre o OTRS para o OTOBO no mesmo servidor e somente então mova a nova instalação.

---

### 5.3 Passo 1: Instalação do novo Sistema OTOBO

Please start with installing a new OTOBO system. Your old OTRS / ((OTRS)) Community Edition installation will be migrated to that new system. We strongly recommend to read the chapter [Instalação do OTOBO](#). For Docker-based installations we refer to the chapter [Instalando usando Docker e Docker Compose](#).

**Aviso:** Under Apache, there are pitfalls with running two independent mod\_perl applications under on the same webserver. Therefore, it is advised to run OTRS and OTOBO on separate webserver. Alternatively remove the OTRS configuration from Apache before installing OTOBO. Use the command `a2query -s` and check the directories /etc/apache2/sites-available and /etc/apache2/sites-enabled for inspecting which configurations are currently available and which are enabled.

After finishing the installation please log in as root@localhost. Navigate to the OTOBO Admin Area Admin -> Packages and install all required OTOBO OPM packages.

**Os seguintes pacotes OPM e OTRS “Feature Addons” NÃO precisa e NÃO deve ser instalado, pois esses recursos já**

- OTRSHideShowDynamicField
- RotherOSSHideShowDynamicField
- TicketForms
- RotherOSS-LongEscalationPerformanceBoost
- Znuny4OTRS-AdvancedDynamicFields
- Znuny4OTRS-AutoSelect
- Znuny4OTRS-EscalationSuspend
- OTRSEscalationSuspend
- OTRSDynamicFieldDatabase
- OTRSDynamicFieldWebService
- OTRSBruteForceAttackProtection
- Znuny4OTRS-ExternalURLJump
- Znuny4OTRS-QuickClose

- Znuny4OTRS-AutoCheckbox
- OTRSSystemConfigurationHistory
- Znuny4OTRS-PasswordPolicy

The following OTOBO packages have been integrated into OTOBO 11.0. This means that they should not be installed in the target system when the target system is OTOBO 11.

- ImportExport

## 5.4 Step 2: Deactivate SecureMode on OTOBO

After installing OTOBO, please log in again to the OTOBO Admin Area Admin -> System Configuration and deactivate the config option SecureMode.

---

**Nota:** Do not forget to actually deploy the changed setting.

---

## 5.5 Step 3: Stop the OTOBO Daemon

This is necessary when the OTOBO Daemon is actually running. Stopping the Daemon is different between Docker-based and non-Docker-based installations.

In the non-Docker case execute the following commands as the user otobo:

```
# in case you are logged in as root
root> su - otobo

otobo> /opt/otobo/bin/Cron.sh stop
otobo> /opt/otobo/bin/otobo.Daemon.pl stop --force
```

When OTOBO is running in Docker, you just need to stop the service daemon:

```
docker_admin> cd /opt/otobo-docker
docker_admin> docker-compose stop daemon
docker_admin> docker-compose ps      # otobo_daemon_1 should have exited with the code 0
↵ 0
```

---

**Nota:** É recomendável executar um backup de todo o sistema OTOBO neste momento. Se algo der errado durante a migração, você não precisará repetir todo o processo de instalação, mas poderá importar o backup para uma nova migração.

**Ver também:**

Nós aconselhamos a ler o capítulo [OTOBO Backup e Restauração](#).

---

## 5.6 Optional Step: Mount /opt/otrs for Convenient Access

Often OTOBO should be running on a new server where /opt/otrs isn't available initially. In these cases the directory /opt/otrs on the OTRS server can be mounted into the file system of the OTOBO server.

When a regular network mount is not possible, then using `sshfs` might be an option.

## 5.7 Optional Step: Install `sshpass` and `rsync` when `/opt/otrs` Should be Copied via `ssh`

This step is only necessary when you want to migrate OTRS from another server and when `/opt/otrs` from the remote server hasn't been mounted on the server running OTOBO.

The tools `sshpass` and `rsync` are needed so that `migration.pl` can copy files via `ssh`. For installing `sshpass`, please log in on the server as user `root` and execute one of the following commands:

```
$ # Install sshpass under Debian / Ubuntu Linux
$ sudo apt-get install sshpass
```

```
$ #Install sshpass under RHEL/CentOS Linux
$ sudo yum install sshpass
```

```
$ # Install sshpass under Fedora
$ sudo dnf install sshpass
```

```
$ # Install sshpass under OpenSUSE Linux
$ sudo zypper install sshpass
```

O mesmo deve ser feito para `rsync` quando ainda não está disponível.

## 5.8 Step 4: Preparing the OTRS / ((OTRS)) Community Edition system

**Nota:** Certifique-se de ter um backup válido do seu sistema OTRS / ((OTRS)) Community Edition também. Sim, não tocamos em nenhum dado OTRS durante a migração, mas às vezes uma entrada incorreta é suficiente para causar problemas.

Agora estamos prontos para a migração. Antes de tudo, precisamos garantir que nenhum ticket seja processado e que nenhum usuário efetue login no OTRS:

Por favor, faça o login na área de administração OTRS Admin -> System Maintenance e adicione um novo slot de manutenção do sistema por algumas horas. Depois disso, exclua todas as sessões de agente e usuário (Admin -> Sessions) e faça logout.

### 5.8.1 Interromper todos os serviços relevantes e o Daemon OTRS

Certifique-se de que não haja serviços ou cron jobs em execução.

```
root> su - otrs
otrs> /opt/otrs/bin/Cron.sh stop
otrs> /opt/otrs/bin/otrs.Daemon.pl stop --force
```

## 5.8.2 Clear the Caches and the Operational Data

The cached data and the operational data doesn't have to be migrated. The mail queue should at this point already be empty.

```
root> su - otrs
otrs> /opt/otrs/bin/otrs.Console.pl Maint::Cache::Delete
otrs> /opt/otrs/bin/otrs.Console.pl Maint::Session::DeleteAll
otrs> /opt/otrs/bin/otrs.Console.pl Maint::Loader::CacheCleanup
otrs> /opt/otrs/bin/otrs.Console.pl Maint::WebUploadCache::Cleanup
otrs> /opt/otrs/bin/otrs.Console.pl Maint::Email::MailQueue --delete-all
```

## 5.9 Optional Step for Docker: make required data available inside container

Existem alguns detalhes a serem considerados quando a instalação do OTOBO está em execução no Docker. O mais relevante: os processos em execução em um contêiner Docker geralmente não podem acessar diretórios fora do contêiner. Porém, há uma exceção: os diretórios montados como volumes no contêiner podem ser acessados. Além disso, observe que o banco de dados MariaDB rodando em `otobo_db_1` não é diretamente acessível de fora da rede de contêiner.

**Nota:** Nos comandos de amostra, presumimos que o usuário `docker_admin` é usado para interagir com o Docker. O administrador do Docker pode ser o usuário `root` do host do Docker ou um usuário dedicado com as permissões necessárias.

### 5.9.1 Copie /opt/otrs dentro do volume `otobo_opt_otobo`

Nesta seção, presumimos que o diretório inicial OTRS `/opt/otrs` está disponível no host Docker.

Existem pelo menos duas possibilidades viáveis:

1. copie `/opt/otrs` para o volume existente `otobo_opt_otobo`
2. montar `/opt/otrs` como um volume adicional

Vamos nos concentrar na opção **a.** aqui.

Primeiro, precisamos descobrir onde o volume `otobo_opt_otobo` está disponível no host Docker.

```
docker_admin> otabo_opt_otobo_mp=$(docker volume inspect --format '{{ .Mountpoint }}' \
↳otobo_opt_otobo)
docker_admin> echo $otobo_opt_otobo_mp # just a sanity check
```

Para uma cópia segura, usamos `rsync`. Dependendo da configuração do Docker, o comando `rsync` pode precisar ser executado com `sudo`.

```
docker_admin> # when docker_admin is root
docker_admin> rsync --recursive --safe-links --owner --group --chown 1000:1000 --
↳perms --chmod "a-wx,Fu+r,Du+rx" /opt/otrs/ $otobo_opt_otobo_mp/var/tmp/copied_otrs
docker_admin> ls -la $otobo_opt_otobo_mp/var/tmp/copied_otrs # just a sanity check

docker_admin> # when docker_admin is not root
```

```
docker_admin> sudo rsync --recursive --safe-links --owner --group --chown 1000:1000 --
↳perms --chmod "a-wx,Fu+r,Du+rx" /opt/otrs/ $otobo_opt_otobo_mp/var/tmp/copied_otrs
docker_admin> sudo ls -la $otobo_opt_otobo_mp/var/tmp/copied_otrs # just a sanity_
↳check
```

Este diretório copiado estará disponível como /opt/otobo/var/tmp/copied\_otrs dentro do contêiner.

## 5.10 Step 5: Perform the Migration!

Please use the web migration tool at <http://localhost/otobo/migration.pl>. Be aware that you might have to replace “localhost” with your OTOBO hostname and you might have to add your non-standard port. The application then guides you through the migration process.

**Aviso:** Às vezes, é mostrado um aviso de que a desativação do **SecureMode** não foi detectada. Reinicie o servidor da web neste caso. Isso força o servidor web a ler a configuração atual.

```
# native installation
root> service apache2 restart

# Docker-based installation
docker_admin> cd /opt/otobo-docker
docker_admin> docker-compose restart web
docker_admin> docker-compose ps # otobo_web_1 should be running again
```

**Nota:** If OTOBO runs inside a Docker container, keep the default settings localhost for the OTRS server and /opt/otobo/var/tmp/copied\_otrs for the OTRS home directory. This is the path of the data that was copied in the optional step.

**Nota:** Os valores padrão para usuário e senha do banco de dados OTRS são obtidos de Kernel/Config.pm no diretório inicial do OTRS. Altere as configurações propostas se estiver usando um usuário de banco de dados dedicado para a migração. Altere também as configurações ao trabalhar com um banco de dados que foi copiado para o contêiner otobo\_db\_1 do Docker.

**Nota:** No caso do Docker, um banco de dados em execução no host Docker não será acessível via 127.0.0.1 de dentro do contêiner Docker. Isso significa que a configuração 127.0.0.1 não será válida para o campo de entrada OTRS Server. Nesse caso, insira um dos endereços IP alternativos relatados pelo comando hostname --all-ip-addresses para o OTRS Server.

**Nota:** Ao migrar para um novo servidor de aplicativos ou para uma instalação baseada em Docker, muitas vezes o banco de dados não pode ser acessado a partir da instalação de destino. Isso geralmente se deve ao fato de que o usuário do banco de dados otobo só pode se conectar a partir do host em que o banco de dados é executado. Para permitir o acesso de qualquer maneira, é recomendável criar um usuário de banco de dados dedicado para a migração. Por exemplo. CREATE USER 'otrs\_migration'@'%' IDENTIFIED BY 'otrs\_migration'; e GRANT

```
SELECT, SHOW VIEW ON otrs.* TO 'otrs_migration'@'%'; Este usuário pode ser descartado novamente após a migração: DROP USER 'otrs_migration'@'%';
```

---

Custom settings in Kernel/Config.pm are carried over from the old OTRS installation to the new OTOBO installation. When you have custom settings, then please take a look at the migrated file /opt/otobo/Kernel/Config.pm. You might want to adapt custom pathes or LDAP settings. In the best case you might find that some custom setting are longer needed.

Quando a migração estiver concluída, não se apresse e teste todo o sistema. Depois de decidir que a migração foi bem-sucedida e que você deseja usar o OTOBO a partir de agora, inicie o Daemon OTOBO:

```
root> su - otobo
otobo>
otobo> /opt/otobo/bin/Cron.sh start
otobo> /opt/otobo/bin/otobo.Daemon.pl start
```

No caso do Docker:

```
docker_admin> cd ~/otobo-docker
docker_admin> docker-compose start daemon
```

## 5.11 Step 6: After Successful Migration!

1. Desinstale sshpass se você não precisar mais.
2. Drop the databases and database users dedicated to the migration if you created any.
3. Divirta-se com o OTOBO!

## 5.12 Known Migration Problems

### 5.12.1 1. Login after migration not possible

Durante nossos testes de migração, o navegador usado para a migração às vezes apresentava problemas. Depois de reiniciar o navegador, esse problema geralmente era resolvido. Com o Safari, às vezes era necessário excluir manualmente a sessão antiga do OTRS.

### 5.12.2 2. Final page of the migration has a strange layout due to missing CSS files

Isso pode acontecer quando a configuração ScriptAlias tem um valor não padrão. A migração simplesmente substitui otrs por otobo. Isso pode fazer com que o CSS e o JavaScript não possam mais ser recuperados no OTOBO. Quando isso acontecer, verifique as configurações em Kernel/Config.pm e reverta-as para os valores normais.

### 5.12.3 3. Migration stops due to MySQL errors

On systems that experienced problems with an upgrade in the past, the migration process may stop due to MySQL errors in the tables ticket and ticket\_history. Usually these errors are NULL values in

the source table that are no longer allowed in the target table. These conflicts have to be manually resolved before you can resume the migration.

As of OTOBO 10.0.12 there is a check in migration.pl that checks for NULL values before the data transfer is done. Note, that the resolution still needs to be performed manually.

#### 5.12.4 4. Errors in Step 5 when migrating to PostgreSQL

In these cases the not so helpful message “System was unable to complete data transfer.” is shown by migration.pl. The Apache logfile, and the OTOBO logfile, show a more meaningful message: “Message: ERROR: permission denied to set parameter “session\_replication\_role”, SQL: ‘set session\_replication\_role to replica;”. In order to give the database user **otobo** the needed superuser privileges, run the following statement as the PostgreSQL admin: ALTER USER otopo WITH SUPERUSER;. Then retry running <http://localhost/otobo/migration.pl>. After the migration, return to the normal state by running ALTER USER otopo WITH NOSUPERUSER.

It is not clear yet, whether the extended privileges have to be granted in every setup.

**Ver também:**

The discussion in <https://otobo.de/de/forums/topic/otrs-6-mysql-migration-to-otobo-postgresql/>.

#### 5.12.5 5. Problems with the Deployment the Merged System Configuration

The system configuration is migrated after the database tables were migrated. In this context, migration means merging the default settings of OTOBO with the system configuration of the source OTRS system. Inconsistencies can arise in this step. An real life example is the setting `Ticket::Frontend::AgentTicketQuickClose###State`. This setting is new in OTOBO 10 and the default value is the state `closed successful`. But this setting is invalid when the state `closed successful` has been dropped or renamed in the source system. This inconsistency is detected as an error in the migration step **Migrate configuration settings**. Actually, the merged system configuration is stored in the database, but additional validity checks are performed during deployment.

The problem must be alleviated manually by using OTOBO console commands.

- List the inconsistencies with the command `bin/otobo.Console.pl Admin::Config::ListInvalid`
- Interactively fix the invalid values with `bin/otobo.Console.pl Admin::Config::FixInvalid`
- Deploy the collected changes from migration.pl, including the deactivated **SecureMode** with `bin/otobo.Console.pl Maint::Config::Rebuild`

After these manual steps you should be able to run migration.pl again. The migration will continue with the step where the error occurred.

## 5.13 Etapa 7: Tarefas de migração manual e alterações

### 5.13.1 1. Password policy rules

Com o OTOBO 10, uma nova política de senha padrão para usuários de agentes e clientes entra em vigor, se a autenticação local for usada. As regras da política de senha podem ser alteradas na configuração do sistema (`PreferencesGroups###Password` and `CustomerPersonalPreference###Password`).

Regra de política de senha	Padrão
PasswordMinSize	8
PasswordMin2Lower2UpperCharacters	Sim
PasswordNeedDigit	Sim
PasswordHistory	10
PasswordTTL	30 dias
PasswordWarnBeforeExpiry	5 dias
PasswordChangeAfterFirstLogin	Sim

### 5.13.2 2. Under Docker: Manually migrate cron jobs

Em uma instalação não Docker do OTOBO, há pelo menos um cron job que verifica a integridade do Daemon. No Docker, esse cron job não existe mais. Além disso, não há daemon cron em execução em nenhum dos contêineres do Docker. Isso significa que você deve procurar uma solução individual para sistemas OTRS com cron jobs específicos do cliente (por exemplo, backup do banco de dados).

## 5.14 Special topics

### 5.14.1 Migration from Oracle to Oracle

For migration to Oracle the ETL-like strategy must be employed. This is because Oracle provides no easy way to temporarily turn off foreign key checks.

On the OTOBO host a Oracle client and the Perl module `DBD::Oracle` must be installed.

---

**Nota:** When using the Oracle instant client, then the optional SDK is also needed for installing `DBD::Oracle`.

---

There are many ways of cloning a schema. In the sample commands we use `expdb` and `impdb` which use Data Pump under the hood.

---

**Nota:** The connect strings shown in this documentation refer to the case when both source and target database run in a Docker container. See also <https://github.com/bschmalhofer/otobo-ideas/blob/master/oracle.md>.

---

1. Clear out otobo

Stop the webserver for otobo, so that the DB connection for otobo is closed.

```
-- in the OTOBO database
DROP USER otobo CASCADE
```

2. Export the complete OTRS schema.

```
mkdir /tmp/otrs_dump_dir
```

```
-- in the OTRS database
CREATE DIRECTORY OTRS_DUMP_DIR AS '/tmp/otrs_dump_dir';
GRANT READ, WRITE ON DIRECTORY OTRS_DUMP_DIR TO sys;
```



```
expdp \"/sys/Oradoc_db1@//127.0.0.1/orclpdb1.localdomain as sysdba\" schemas=otrs_
↳directory=OTRS_DUMP_DIR dumpfile=otrs.dmp logfile=expdpotrs.log
```

3. Import the OTRS schema, renaming the schema to 'otobo'.

```
impdp \"/sys/Oradoc_db1@//127.0.0.1/orclpdb1.localdomain as sysdba\" directory=OTRS_
↳DUMP_DIR dumpfile=otrs.dmp logfile=impdpotobo.log remap_schema=otrs:otobo
```

```
-- in the OTOBO database
-- double check
select owner, table_name from all_tables where table_name like 'ARTICLE_DATA_OT%CHAT
↳';

-- optionally, set the password for the user otobo
ALTER USER otobo IDENTIFIED BY XXXXXX;
```

4. Adapt the cloned schema otobo

```
cd /opt/otobo
scripts/backup.pl --backup-type migratefromotrs # it's OK that the command knows only_
↳about the otobo database, only last line is relevant
sqlplus otobo/otobo@//127.0.0.1/orclpdb1.localdomain < /home/bernhard/devel/OTOBO/
↳otobo/2021-03-31_13-36-55/orclpdb1.localdomain_post.sql >sqlplus.out 2>&1
double check with `select owner, table_name from all_tables where table_name like
↳'ARTICLE_DATA_OT%CHAT';
```

5. Start the web server for otobo again

6. Proceed with step 5, that is with running migration.pl.

---

**Nota:** If migrating to OTOBO version greater or equal 10.1 the script `/opt/otobo/scripts/DBUpdate-to-10.1.pl` has to be executed, to create the tables `stats_report` & `data_storage`, which were newly added in version 10.1.

---

## 5.14.2 Optional Step: Streamlined migration of the database (only for experts and special scenarios)

In the general migration strategy, all data in the database tables is copied row by row from the OTRS database into the OTOBO database. Exporting the data from the OTRS database and importing it into the OTOBO database might save time and is more stable in some circumstances.

---

**Nota:** This variant works for both Docker-based and native installations.

---

**Nota:** These instructions assume that OTRS is using MySQL as its backend.

---

First of all, we need a dump of the needed OTRS database tables. Then we need to perform a couple of transformations:

- convert the character set to utf8mb4
- rename a couple of tables

- shorten some table columns

After the transformation we can overwrite the tables in the OTOBO schema with the transformed data from OTRS. Effectively we need not a single dump file, but several SQL scripts.

Quando o `mysqldump` é instalado e uma conexão com o banco de dados OTRS é possível, você pode criar o dump do banco de dados diretamente no host Docker. Este caso é suportado pelo script `bin/backup.pl`.

**Aviso:** Isso requer que uma instalação OTOBO esteja disponível no host Docker.

```
otobo> cd /opt/otobo
otobo> scripts/backup.pl -t migratefromotrs --db-name otrs --db-host=127.0.0.1 --db-
↪user otrs --db-password "secret_otrs_password"
```

---

**Nota:** Alternatively, the database can be dumped on another server and then be transferred to the Docker host afterwards. An easy way to do this is to copy `/opt/otobo` to the server running OTRS and perform the same command as above.

---

The script `bin/backup.pl` generates four SQL scripts in a dump directory, e.g. in `2021-04-13_12-13-04` In order to execute the SQL scripts, we need to run the command `mysql`.

Native installation:

```
otobo> cd <dump_dir>
otobo> mysql -u root -p<root_secret> otabo < otrs_pre.sql
otobo> mysql -u root -p<root_secret> otabo < otrs_schema_for_otobo.sql
otobo> mysql -u root -p<root_secret> otabo < otrs_data.sql
otobo> mysql -u root -p<root_secret> otabo < otrs_post.sql
```

Docker-based installation:

Run the command `mysql` within the Docker container `db` for importing the database dump files. Note that the password for the database root is now the password that has been set up in the file `.env` on the Docker host.

```
docker_admin> cd /opt/otobo-docker
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otabo < /opt/
↪otobo/<dump_dir>/otrs_pre.sql
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otabo < /opt/
↪otobo/<dump_dir>/otrs_schema_for_otobo.sql
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otabo < /opt/
↪otobo/<dump_dir>/otrs_data.sql
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otabo < /opt/
↪otobo/<dump_dir>/otrs_post.sql
```

Para uma verificação rápida se a importação funcionou, você pode executar os seguintes comandos.

```
otobo> mysql -u root -p<root_secret> -e 'SHOW DATABASES'
otobo> mysql -u root -p<root_secret> otabo -e 'SHOW TABLES'
otobo> mysql -u root -p<root_secret> otabo -e 'SHOW CREATE TABLE ticket'
```

or when running under Docker

```
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> -e 'SHOW
↳DATABASES'
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otobo -e 'SHOW
↳TABLES'
docker_admin> docker-compose exec -T db mysql -u root -p<root_secret> otobo -e 'SHOW
↳CREATE TABLE ticket'
```

The database is now migrated. This means that during the next step we can skip the database migration. Watch out for the relevant checkbox.



## Atualizando

---

**Nota:** É altamente recomendável realizar uma atualização de teste em uma máquina de teste separada primeiro.

---

**Nota:** On Debian systems you may need to manually install some perl packages before upgrading from 10.0 to 10.1

```
apt-get install -y libarchive-zip-perl libtimedate-perl libdatettime-perl
↳libconvert-binhex-perl libcgi-psgi-perl libdbi-perl libdbix-connector-perl
↳libfile-chmod-perl liblist-allutils-perl libmoo-perl libnamespace-autoclean-
↳perl libnet-dns-perl libnet-smtp-ssl-perl libpath-class-perl libsub-
↳exporter-perl libtemplate-perl libtemplate-perl libtext-trim-perl libtry-
↳tiny-perl libxml-libxml-perl libyaml-libyaml-perl libdbd-mysql-perl
↳libapache2-mod-perl2 libmail-imapclient-perl libauthen-sasl-perl libauthen-
↳ntlm-perl libjson-xs-perl libtext-csv-xs-perl libpath-class-perl libplack-
↳perl libplack-middleware-header-perl libplack-perl libplack-middleware-
↳reverseproxy-perl libencode-hanextra-perl libio-socket-ssl-perl libnet-
↳ldap-perl libcrypt-eksblowfish-perl libxml-libxslt-perl libxml-parser-perl
↳libconst-fast-perl
```

## 6.1 Etapa 1: interromper todos os serviços relevantes e o OTOBO Daemon

Certifique-se de que não haja mais serviços em execução ou tarefas cron que tentem acessar o OTOBO. Isso dependerá da configuração do seu serviço.

```
root> systemctl stop postfix
root> systemctl stop apache2
root> systemctl stop cron
```

Pare os cron jobs do OTOBO e o daemon (nesta ordem):

```
root> su - otobo
otobo> cd /opt/otobo/
otobo> bin/Cron.sh stop
otobo> bin/otobo.Daemon.pl stop
```

## 6.2 Etapa 2: Arquivos de backup e banco de dados

Crie um backup do diretório /opt/otobo e do banco de dados.

### 6.2.1 Exemplo de instalação padrão com Ubuntu e MySQL

```
root> mkdir /root/otobo-update # Create a update directory
root> cd /root/otobo-update # Change into the update directory
root> cp -pr /opt/otobo otobo-prod-old # Backup the hole OTOBO directory
↳to the update directory
root> mysqldump -u otobo -p otobo -r otobo-prod-old.sql # Backup the otobo database
↳to otobo-prod-old.sql
```

Verifique se todos os arquivos são válidos. Agora temos um backup com todos os dados necessários.

**Aviso:** Não continue sem um backup completo do seu sistema. Você também pode usar o script: backup-restore para isso.

## 6.3 Etapa 3: instalar a nova versão

Baixe a última versão do otobo em <https://ftp.otobo.org/pub/otobo/>. e descompacte o arquivo fonte (por exemplo, usando "tar") dentro do diretório /root/otobo-update:

```
root> cd /root/otobo-update # Change into
↳the update directory
root> wget https://ftp.otobo.org/pub/otobo/otobo-latest-10.1.tar.gz # Download he
↳latest OTOBO 10.1 release
root> tar -xzf otobo-latest-10.1.tar.gz # Unzip OTOBO
root> cp -r otobo-10.1.x/* /opt/otobo # Copy the
↳new otobo directory to /opt/otobo
```

### 6.3.1 Restaurar arquivos de configuração antigos

Precisamos apenas copiar o arquivo "Kernel/Config.pm" no OTOBO 10.

```
root> cd /root/otobo-update
root> cp -p otopo-prod-old/Kernel/Config.pm /opt/otobo/Kernel/
root> cp -p otopo-prod-old/var/cron/* /opt/otobo/var/cron/
```

### 6.3.2 Restaurar dados do artigo

Se você configurou o OTOBO para armazenar dados do artigo no sistema de arquivos, você deve restaurar a pasta `article` para `/opt/otobo/var/` ou a pasta especificada na configuração do sistema.

```
root> cd /root/otobo-update
root> cp -pr otopo-prod-old/var/article/* /opt/otobo/var/article/
```

### 6.3.3 Restaurar estatísticas padrão já instaladas

Se você tiver pacotes adicionais com estatísticas padrão, você deve restaurar os arquivos XML de estatísticas com o sufixo `*.installed` para `/opt/otobo/var/stats`.

```
root> cd /root/otobo-update/otobo-prod-old/var/stats
root> cp *.installed /opt/otobo/var/stats
```

### 6.3.4 Definir permissões de arquivo

Por favor, execute o seguinte comando para definir as permissões de arquivo e diretório para o OTOBO. Ele tentará detectar as configurações corretas de usuário e grupo necessárias para sua configuração.

```
root> /opt/otobo/bin/otobo.SetPermissions.pl
```

### 6.3.5 Check Apache configuration files

Newer versions of OTOBO may need you to adjust the apache configuration. From version 10.1 and onwards we moved from CGI to PSGI. Take a look at `scripts/apache2-httpd-vhost-443.include.conf` to see what settings needs to be adjusted/added.

## 6.4 Step 4: Check for new needed perl modules

OTOBO needs new cpan packages for some version jumps. Please check if new packages are needed and install them if necessary.

**Nota:** On Debian systems you may need to manually install some packages:

```
apt-get install -y libarchive-zip-perl libtimedate-perl libdatettime-perl libconvert-
↳ binhex-perl libcgi-psgi-perl libdbi-perl libdbix-connector-perl libfile-chmod-perl
↳ liblist-allutils-perl libmoo-perl libnamespace-autoclean-perl libnet-dns-perl
↳ libnet-smtp-ssl-perl libpath-class-perl libsub-exporter-perl libtemplate-perl
↳ libtemplate-perl libtext-trim-perl libtry-tiny-perl libxml-libxml-perl libyaml-
↳ libyaml-perl libdbd-mysql-perl libapache2-mod-perl2 libmail-imapclient-perl
↳ libauthen-sasl-perl libauthen-ntlm-perl libjson-xs-perl libtext-csv-xs-perl libpath-
↳ class-perl libplack-perl libplack-middleware-header-perl libplack-perl libplack-
↳ middleware-reverseproxy-perl libencode-hanextra-perl libio-socket-ssl-perl libnet-
↳ ldap-perl libcrypt-eksblowfish-perl libxml-libxslt-perl libxml-parser-perl libconst-
↳ fast-perl
```

### 6.4. Step 4: Check for new needed perl modules

```
root> su - otobo
otobo> perl /opt/otobo/bin/otobo.CheckModules.pl --list
```

## 6.5 Step 5: Update Installed Packages and reconfigure config

Você pode usar o comando abaixo para atualizar todos os pacotes instalados. Isso funciona para todos os pacotes disponíveis em repositórios online. Você pode atualizar outros pacotes mais tarde através do gerenciador de pacotes (isso requer um daemon OTOBO em execução).

```
root> su - otobo
otobo> /opt/otobo/bin/otobo.Console.pl Admin::Package::ReinstallAll
otobo> /opt/otobo/bin/otobo.Console.pl Admin::Package::UpgradeAll
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Config::Rebuild
```

## 6.6 Step 6: Only for minor or major release upgrades (for example to upgrade from 10.0 to 10.1)

```
root> su - otobo
otobo> /opt/otobo/scripts/DBUpdate-to-10.1.pl
```

## 6.7 Step 7: Start your Services

Start OTOBO cron jobs and the daemon (in this order):

```
root> su - otobo
otobo> cd /opt/otobo/
otobo> bin/otobo.Daemon.pl start
otobo> bin/Cron.sh start
```

Agora os serviços podem ser iniciados. Isso vai depender da configuração do seu serviço, aqui está um exemplo:

```
root> systemctl start postfix
root> systemctl start apache2
root> systemctl start cron
```

Agora você pode entrar em seu sistema.



---

## Atualizando uma instalação baseada em Docker do OTOBO

---

Para executar o OTOBO no Docker, precisamos do próprio software OTOBO e de um ambiente no qual o OTOBO possa ser executado. A imagem OTOBO Docker fornece o ambiente e uma cópia do software OTOBO. O próprio software é instalado no volume `otobo_opt_otobo`. Um volume nomeado é usado porque os dados de tempo de execução, por exemplo, arquivos de configuração e pacotes instalados, são armazenados na mesma árvore de diretório.

Ao atualizar para uma nova versão do OTOBO, várias coisas precisam acontecer.

- Os arquivos Docker Compose devem ser atualizados.
- O arquivo de configuração `.env` do Docker Compose deve ser verificado.
- A nova imagem do Docker deve ser buscada.
- O volume `otobo_opt_otobo` deve ser atualizado.
- Algumas tarefas de manutenção devem ser executadas.

---

**Nota:** In the sample commands below, the version **11.x.y**, corresponding to the tag **11\_x\_y**, is used as the example version. Please substitute it with the real version, e.g. **11.0.7**.

---

### 7.1 Atualizar os arquivos Docker Compose

Os arquivos OTOBO Docker Compose podem mudar entre as versões. Portanto, deve-se ter certeza de que a configuração correta é usada.

---

**Nota:** See <https://hub.docker.com/r/rotheross/otobo/tags> for the available releases.

---

```
# Change to the otobo docker directory
docker_admin> cd /opt/otobo-docker
```

```
# Get the latest tags
docker-admin> git fetch --tags

# Update OTOBO docker-compose repository to version 11.x.y.
docker-admin> git checkout rel-11_x_y
```

## 7.2 Verificando o arquivo Docker Compose .env

O arquivo `.env` controla o contêiner do Docker OTOBO. Nesse arquivo, as variáveis `OTOBO_IMAGE_OTOBO`, `OTOBO_IMAGE_OTOBO_ELASTICSEARCH`, e `OTOBO_IMAGE_OTOBO_NGINX` declaram quais imagens são usadas. As imagens mais recentes são usadas quando essas variáveis não são definidas. Se você quiser usar uma versão específica, defina essas variáveis de acordo.

## 7.3 Busque as novas imagens do Docker

Docker compose can be used for fetching the wanted images from <https://hub.docker.com/r/rotheross/otobo/>.

```
# Change to the otopo docker directory
docker_admin> cd /opt/otobo-docker

# fetch the new images, either 'latest-11_0', 'latest-10_1', 'latest-10_0' or the_
↪specific version declared in .env
docker_admin> docker-compose pull
```

## 7.4 Atualize o OTOBO

**Aviso:** Please note that minor or major upgrades must always be carried out one after the other. If you would like to upgrade from version 10.0.\* to the latest 11.0.\*, please upgrade to 10.1 first and then to 11.0.

Nesta etapa, o volume `otobo_opt_otobo` é atualizado e os seguintes comandos do console OTOBO são executados:

- Admin::Package::ReinstallAll
- Admin::Package::UpgradeAll
- Maint::Config::Rebuild
- Maint::Cache::Delete

```
# stop and remove the containers, but keep the named volumes
docker_admin> docker-compose down

# copy the OTOBO software, while containers are still stopped
docker_admin> docker-compose run --no-deps --rm web copy_otobo_next
```

```
# start containers again, using the new version and the updated /opt/otobo
docker_admin> docker-compose up --detach

# a quick sanity check
docker_admin> docker-compose ps

# complete the update, with running database
docker_admin> docker-compose exec web /opt/otobo_install/entrypoint.sh do_update_tasks

# inspect the update log
docker_admin> docker-compose exec web cat /opt/otobo/var/log/update.log

**# For minor or major release upgrades, you also have to run the upgrade script (for
↳example to upgrade from 10.1 to 11.0)**
root> docker exec -it otobo_web_1 perl scripts/DBUpdate-to-11.0.pl
```

**Nota:** Os comandos listados acima podem ser automatizados. Para tal, o script `scripts/update.sh` será disponibilizado no OTOBO 10.0.8. Este script executa os comandos, começando com o comando **docker-compose pull**.

```
docker_admin> ./scripts/update.sh --help
docker_admin> ./scripts/update.sh

** For minor or major release upgrades, you also have to run the upgrade script (for
↳example to upgrade from 10.1 to 11.0)**
docker_admin> docker exec -it otobo_web_1 perl scripts/DBUpdate-to-11.0.pl
```



---

## Backup e Restauração

---

OTOBO possui scripts internos para backup e restauração. Execute os scripts com a opção `-h` para obter mais informações.

### 8.1 Backup

**Nota:** Para criar um backup, é necessário permissão de gravação para o usuário `otobo` no diretório de destino.

```
otobo> /opt/otobo/scripts/backup.pl -h
```

Saída do script:

```
Backup an OTOBO system.

Usage:
  backup.pl -d /data_backup_dir [-c gzip|bzip2] [-r DAYS] [-t
↪fullbackup|nofullbackup|dbonly]
  backup.pl --backup-dir /data_backup_dir [--compress gzip|bzip2] [--remove-old-
↪backups DAYS] [--backup-type fullbackup|nofullbackup|dbonly]

Short options:
  [-h]                - Display help for this command.
  -d                  - Directory where the backup files should place to.
  [-c]                - Select the compression method (gzip|bzip2). Default: gzip.
  [-r DAYS]           - Remove backups which are more than DAYS days old.
  [-t]                - Specify which data will be saved
↪(fullbackup|nofullbackup|dbonly). Default: fullbackup.

Long options:
```

```

[--help]                - same as -h
--backup-dir            - same as -d
[--compress]           - same as -c
[--remove-old-backups DAYS] - same as -r
[--backup-type]        - same as -t
    
```

Help:

Using `-t fullbackup` saves the database and the whole OTOBO home directory (except `/var/tmp` and cache directories).

Using `-t nofullbackup` saves only the database, `/Kernel/Config*` and `/var` directories. With `-t dbonly` only the database will be saved.

Override the max allowed packet size:

When backing up a MySQL one might run into very large database fields. In this case, the backup fails.

For making the backup succeed one can explicitly add the parameter `--max-allowed-packet=<SIZE IN BYTES>`.

This setting will be passed on to the command `mysqldump`.

Output:

```

Config.tar.gz          - Backup of /Kernel/Config* configuration files.
Application.tar.gz     - Backup of application file system (in case of full backup).
VarDir.tar.gz         - Backup of /var directory (in case of no full backup).
DataDir.tar.gz        - Backup of article files.
DatabaseBackup.sql.gz - Database dump.
    
```

## 8.2 Restaurar

**Nota:** Para restaurar o banco de dados, certifique-se de que o banco de dados `otobo` existe e não contém tabelas.

```
otobo> /opt/otobo/scripts/restore.pl -h
```

Saída do script:

```
Restore an OTOBO system from backup.
```

Usage:

```
restore.pl -b /data_backup/<TIME>/ -d /opt/otobo/
```

Options:

```

-b                - Directory of the backup files.
-d                - Target OTOBO home directory.
[-h]             - Display help for this command.
    
```

## 8.3 Considerações para executar OTOBO no Docker

Os mesmos scripts podem ser usados com OTOBO em execução no Docker. No entanto, algumas limitações específicas do Docker devem ser consideradas.

Primeiro, precisamos nos certificar de que os arquivos de backup não sejam criados no sistema de arquivos interno do contêiner. Porque nesse caso todos os dados seriam perdidos quando o contêiner fosse interrompido. Portanto, o diretório de backup deve estar em um volume. Por enquanto, consideramos apenas o caso mais simples, em que o diretório de backup é um diretório local no host Docker. A localização do diretório de backup no contêiner pode ser escolhida arbitrariamente. Neste exemplo, escolhemos o diretório local `otobo_backup` como o local no host, e `/otobo_backup` como o local no contêiner.

Primeiro, precisamos criar o volume.

```
# create the backup directory on the host
docker_admin> mkdir otopo_backup

# create the Docker volume
docker_admin> docker volume create --name otopo_backup --opt type=none --opt device=
↳$PWD/otobo_backup --opt o=bind

# inspect the volume out of curiosity
docker_admin> docker volume inspect otopo_backup
```

Para criar o backup, precisamos de um banco de dados em execução e os volumes `otobo_opt_otobo` e `otobo_backup`. Isso significa que o servidor da Web e o Daemon podem, mas não precisam, ser interrompidos.

```
# create a backup
docker_admin> docker run -it --rm --volume otopo_opt_otobo:/opt/otobo --volume otopo_
↳backup:/otobo_backup --network otopo_default rotheross/otobo:latest-10_0 scripts/
↳backup.pl -d /otobo_backup

# check the backup file
docker_admin> tree otopo_backup
```

Para restaurar o backup, também precisamos especificar qual backup deve ser restaurado. O placeholder `<TIMESTAMP>` é algo como `2020-09-07_09-38`.

```
# create a backup
docker_admin> docker run -it --rm --volume otopo_opt_otobo:/opt/otobo --volume otopo_
↳backup:/otobo_backup --network otopo_default rotheross/otobo:latest-10_0 scripts/
↳restore.pl -d /opt/otobo -b /otobo_backup/<TIMESTAMP>
```





---

## Backup e Restauração usando Docker

---

Por favor, leia o capítulo [Backup e Restauração](#) para informações básicas sobre os scripts de backup e restauração.

### 9.1 Considerações para executar OTOBO no Docker

Os scripts padrão `backup.pl` e `restore.pl` também podem ser usados com OTOBO rodando no Docker. No entanto, algumas limitações específicas do Docker devem ser consideradas.

Primeiro, precisamos nos certificar de que os arquivos de backup não sejam criados no sistema de arquivos interno de um contêiner do Docker. Porque, nesse caso, os arquivos criados seriam perdidos quando o contêiner for interrompido. Isso significa que o diretório de backup deve estar localizado em um volume. Para este manual, consideramos apenas o caso mais simples, em que o diretório de backup é um diretório local no host Docker. A localização do diretório de backup no contêiner pode ser escolhida arbitrariamente. Neste exemplo, escolhemos o dir local `otobo_backup` como o local no host e `/otobo_backup` como o local no contêiner.

Em segundo lugar, os comandos no contêiner Docker geralmente são executados como o usuário `otobo` com a id de usuário 1000 e a id de grupo 1000. Deve-se ter certeza de que este usuário pode escrever no diretório de backup.

Primeiro, precisamos criar o volume.

```
# create the backup directory on the host
docker_admin>mkdir otopo_backup

# give the backup dir to the user otopo, elevated privs might be needed for that
docker_admin>chown 1000:1000 otopo_backup

# create the Docker volume
docker_admin>docker volume create --name otopo_backup --opt type=none --opt device=
↳$PWD/otobo_backup --opt o=bind
```

```
# inspect the volume out of curiosity
docker_admin>docker volume inspect otobo_backup
```

Para criar o backup, precisamos de um banco de dados em execução e os volumes `otobo_opt_otobo` e `otobo_backup`. Isso significa que o servidor da web e o daemon OTOBO podem, mas não precisam, ser parados.

```
# create a backup
docker_admin>docker run -it --rm --volume otobo_opt_otobo:/opt/otobo --volume otobo_
↳backup:/otobo_backup --network otobo_default rotheross/otobo:latest-10_0 scripts/
↳backup.pl --extra-dump-options="--single-transaction" -d /otobo_backup

# check the backup file
docker_admin>tree otobo_backup

.. note::

--extra-dump-options="--single-transaction" prevents the database tables from being
↳locked, so OTOBO can still be used during the backup.
```

---

**Nota:** Para restaurar o banco de dados, certifique-se de que o banco de dados `otobo` existe e não contém tabelas.

---

To drop an existing `otobo` database and create a new one you can use the following commands. First, you have to connect to the MySQL CLI of the db container.

As soon as you are connected to the MySQL server, you can drop and recreate the `otobo` database.

```
mysql@4f7783595190:/$>DROP DATABASE otobo;
mysql@4f7783595190:/$>CREATE DATABASE otobo CHARACTER SET utf8mb4 COLLATE utf8mb4_
↳unicode_ci;
mysql@4f7783595190:/$>GRANT ALL PRIVILEGES ON otobo.* TO 'otobo'@'%';
```

Para restaurar o backup, também precisamos especificar qual backup deve ser restaurado. O placeholder `<TIMESTAMP>` é algo como `2020-09-07_09-38`.

```
# restore a backup
docker_admin>docker run -it --rm --volume otobo_opt_otobo:/opt/otobo --volume otobo_
↳backup:/otobo_backup --network otobo_default rotheross/otobo:latest-10_0 scripts/
↳restore.pl -d /opt/otobo -b /otobo_backup/<TIMESTAMP>
```

---

## Kerberos Single Sign On in OTOBO Docker installation

---

Please read to the chapter [Instalando usando Docker e Docker Compose](#) for basic information about installing and configure OTOBO. This tutorial assumes that OTOBO has been installed and configured using Docker.

---

**Nota:** In the following, we will refer from AD (Active Directory), of course the Kerberos configuration is also possible under LDAP.

---

### 10.1 Generate Active Directory User

Please create a new Active Directory User with the following settings and save the marked settings:

---

**Nota:** Please use as Username only this syntax: HTTP/fqdn.from.your.otobo.de. fqdn.from.your.otobo.de needs to be a A-Record DNS entry, not a CNAME! In the next step, it is also possible to use other URLs for OTOBO, they must then point as CNAME to our A-record defined above.

The username part “HTTP/” should be written in capital letters, as Kerberos expects it that way.

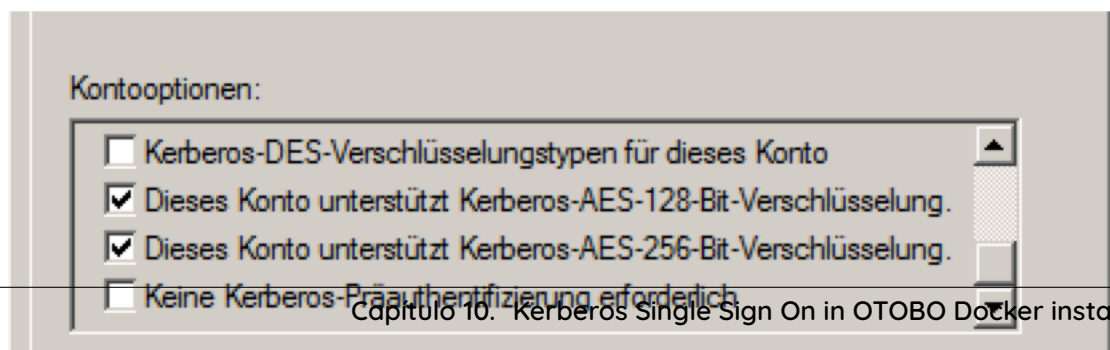
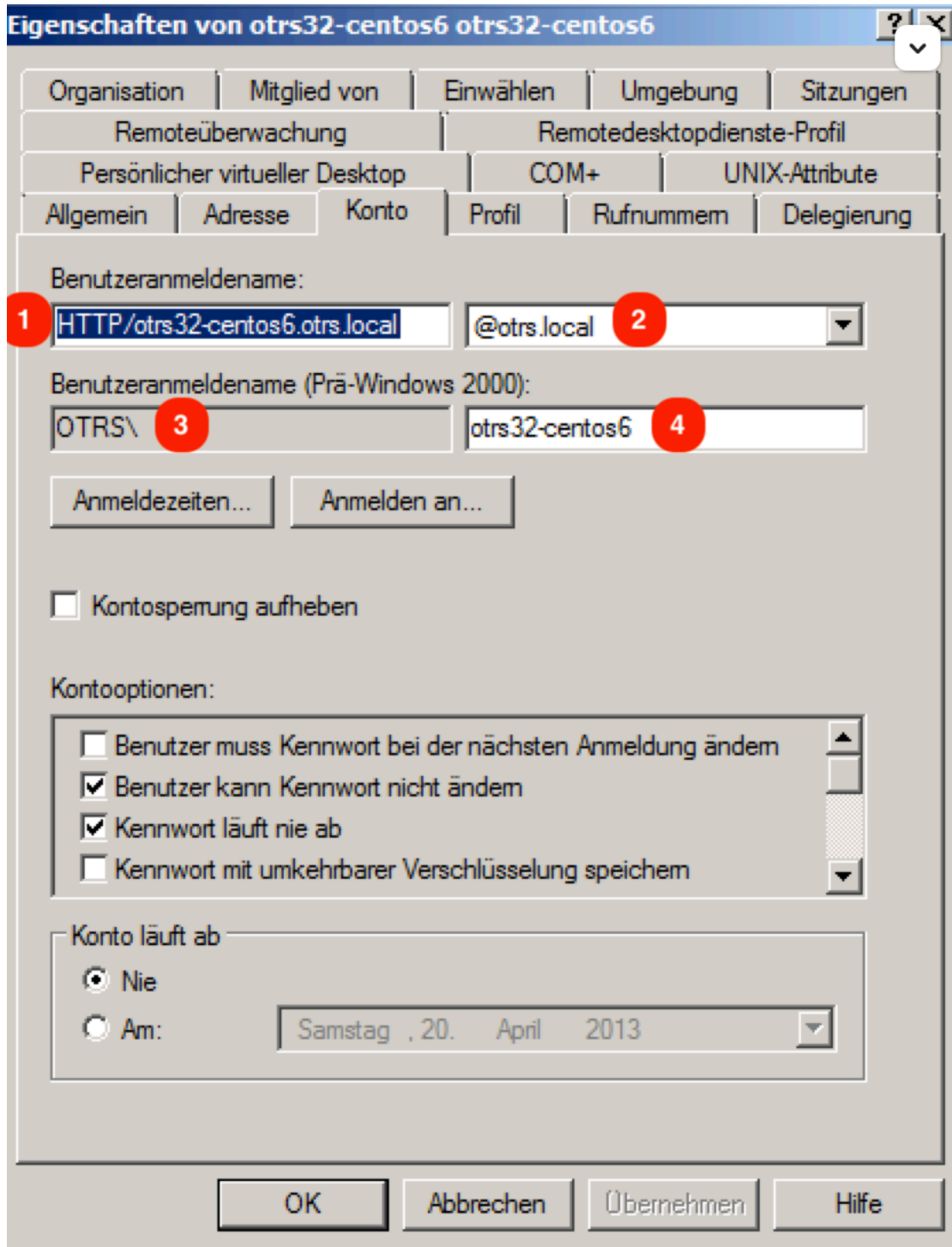
The password doesn't work properly with some special characters (e.g. '&').

You have to create a separate AD-user. You can not use the one that you already use for your LDAP/AD sync.

---

### 10.2 Generate Active Directory Keytab file

In the next step, we connect to a domain controller of the Active Directory and open a console (cmd) there with administrator privileges. Now we use the tool 'ktpass.exe' to generate the needed keytab file:



```
ktpass.exe -princ HTTP/otrs32-centos6.otrs.local@OTRS.LOCAL -mapuser OTRS\otrs32-
↪centos6 -crypto All -pass Password -ptype KRB5_NT_PRINCIPAL -out c:\krb5.keytab
```

- -princ = HTTP/otrs32-centos6.otrs.local@OTRS.LOCAL -> Picture Number 1+@+Picture Number 2
- -mapuser = OTRSotrs32-centos6 (Username prä Win 2000) -> -> Picture Number 3++Picture Number
- -pass = Password from user otrs32-centos6 (Active Directory User)
- -out = c:/krb5.keytab

**Nota:** Please write the domain (@OTRS.LOCAL) always in capital letters. The password must not contain some special characters.

In the next step please move the krb5.keytab file to the OTOBO Server:

```
# Create new directory
docker_admin> mkdir /opt/otobo-docker/nginx-conf

# Move the file krb5.keytab to the new directory (Attention, depending on where you
↪have placed the krb5.conf file, the command below will change.)
docker_admin> mv ?/krb5.keytab /opt/otobo-docker/nginx-conf/krb5.keytab
```

## 10.3 Create a new volume for your custom nginx configuration

```
docker volume create otobo_nginx_custom_config
otobo_nginx_custom_config_mp=$(docker volume inspect --format '{{ .Mountpoint }}'
↪otobo_nginx_custom_config)
docker create --name tmp-nginx-container rotheross/otobo-nginx-webproxy:latest-10_1
↪(achtung: Versionsnummer)
docker cp tmp-nginx-container:/etc/nginx/templates /tmp
docker cp tmp-nginx-container:/etc/nginx/templates/otobo_nginx-kerberos.conf.template.
↪hidden $otobo_nginx_custom_config_mp/otobo_nginx.conf.template
docker rm tmp-nginx-container
vim docker-compose/otobo-nginx-custom-config.yml
```

```
COMPOSE_FILE =>
docker-compose/otobo-nginx-custom-config.yml
NGINX_ENVSUBST_TEMPLATE_DIR=/etc/nginx/config/template-custom
```

## 10.4 Create new OTOBO .env file

First of all we need to move the old file /opt/otobo-docker/.env to .env.tmp and create a new .env file including the kerberos settings.

```
# Stop OTOBO Container if running
docker_admin>cd /opt/otobo-docker
docker_admin>docker-compose down

# create a backup of the old .env file
```

```
docker_admin>mv /opt/otobo-docker/.env /opt/otobo-docker/.env.tmp

# create a new backupfile including kerberos settings
docker_admin>cp /opt/otobo-docker/.docker_compose_env_https_kerberos /opt/otobo-
↪docker/.env
```

Now copy your existing configuration options to the new .env file (at least OTOBO\_DB\_ROOT\_PASSWORD, OTOBO\_NGINX\_SSL\_CERTIFICATE, OTOBO\_NGINX\_SSL\_CERTIFICATE\_KEY) and insert the following Kerberos settings:

```
# Kerberos keytab OTOBO_NGINX_KERBEROS_KEYTAB=/opt/otobo-docker/nginx-conf/krb5.keytab
# Kerberos config (Important, please comment out this option like here!) # In default configuration
the krb5.conf file is generated automatically # OTOBO_NGINX_KERBEROS_CONFIG=/opt/otobo-
docker/nginx-conf/krb5.conf

# Kerberos Service Name OTOBO_NGINX_KERBEROS_SERVICE_NAME=HTTP/otrs32-
centos6.otrs.local # -> Picture Number 1

# Kerberos REALM OTOBO_NGINX_KERBEROS_REALM=ROTHER-OSS.COM -> OTRS.LOCAL # ->
Picture Number 2

# Active Directory Domain Controller / Kerberos kdc OTOBO_NGINX_KERBEROS_KDC=
# Active Directory Domain Controller / Kerberos Admin Server OTOBO_NGINX_KERBEROS_ADMIN_SERVER=rother-
oss.com

# Kerberos Default Domain OTOBO_NGINX_KERBEROS_DEFAULT_DOMAIN=otrs.local
```

## 10.5 Start OTOBO

After the initial Kerberos configuration we start OTOBO again:

```
# Start OTOBO using docker-compose
docker_admin> docker-compose up -d
```

## 10.6 Tell OTOBO to use the Kerberos-Authentication

In case you have configured AD-Authentication, de-activate it (e.g. by commenting out the respective lines from your Kernel/Config.pm). The authentication will not take place via LDAP anymore.

To use Kerberos-Authentication take the Kerberos-lines from Kernel/Config/Defaults.pm and put it into you Kernel/Config.pm E.g. these lines could work:

```
$self->{AuthModule} = 'Kernel::System::Auth::HTTPBasicAuth';

# In case you need to replace some part of the REMOTE_USER, you can
# use the following RegExp ($1 will be new login).
$self->{'AuthModule::HTTPBasicAuth::ReplaceRegExp'} = '^(.+?)@.+?$';
```

## 10.7 Configure Browser to understand Kerberos SSO

For SSO to work, the browser must be configured accordingly.

**Chrome, Edge, Internet Explorer, etc.**

Add page under local or trusted pages and activate 'Integrated Windows Authentication' (Internet Options).

**Firefox**

Enter "about:config" in the firefox address line

and change the following settings:

- network.negotiate-auth.trusted-uris = https:// (or https://otobofqdn)
- network.negotiate-auth.delegation-uris = http:// (or https://otobofqdn)

## 10.8 Debugging and Problems

If the Kerberos SSO does not work, please check first if the NGINX container is started:

```
# Check Container
docker_admin> docker ps
```

In the next step please check the NGINX logs for more information:

```
# Check NGINX logs
docker_admin> docker logs otobo_nginx_1 -f
```

If NGINX is running, please login into the NGINX Container and check all needed files:

```
# Login to the NGINX Container
docker_admin> docker exec -it otobo_nginx_1 bash

# Now please check if the krb5.conf file exists with your needed values
nginx_root> cat /etc/krb5.conf

# Now please check if the krb5.keytab file exists
nginx_root> cat /etc/krb5.keytab

# If not, please quit from the container and copy the file again using docker
docker_admin> docker cp /opt/otobo-docker/nginx-conf/krb5.keytab otobo_nginx_1:/etc/
↵krb5.keytab
```

### 10.8.1 Kerberos debugging

```
# Login to the NGINX Container
docker_admin> docker exec -it otobo_nginx_1 bash
```

Now you are able to debug the Kerberos settings. Examples:

```
env KRB5_TRACE=/dev/stdout kvno HTTP/otrs32-centos6.otrs.local@OTRS.LOCAL
klist -e
```

```
kinit -VV -k -t /etc/krb5.keytab HTTP/otrs32-centos6.otrs.local@OTRS.LOCAL
```

In case you stumble upon the issue that apparently the authentication works but the agent is not yet in the database, then your sync (if implemented) might not work. An error 52e (First bind failed) indicates that something is wrong with your Search User. This happens if you use the same user for the AD sync and as a SSO user. Please use separate AD users for that. In order to not have to create a new keytab and having to repeat the steps mentioned above, it could be easier to create a new user to use in your AD sync (probably in your Kernel/Config.pm).

In case SSO is not working properly, make sure: \* the user for which it is not working is in Active Directory \* the system has to be in the domain \* it is properly stated as a trusted page (see 'Configure Browser to understand Kerberos SSO')



---

## Adapt customer interface with corporate identity

---

In OTOBO it is very easy to adapt the customer area to your own corporate identity. Follow this tutorial step by step and OTOBO will shine in your own design in a short time.

---

**Nota:** Currently, it is not so easy to adapt the agent area to one's own CI. Changes in the OTOBO .css files would be necessary here. One exception is the logo on the agent login page and the agent header. The logos can be easily exchanged by copying the logos to the server and then adjusting the options `AgentLoginLogo` and `AgentLogo` under `Admin -> System Configuration`.

---

### 11.1 Change colors in Customer Area

To change the colors for the OTOBO customer interface, please go to `Admin -> System Configuration` and change the following settings:

- `CustomerColorDefinitions`
- To change the colours on the Customer Dashboard, please go to `Admin -> System Configuration` and search for `CustomerDashboard`. In the search result you will find all the options you need with colour definitions.

### 11.2 Change Logos and Pictures

In the first step please copy your Logos and Pictures to the OTOBO Server. Please use an SCP client (WinSCP) for this purpose. Often you do not have the permissions to copy the logos to the right place. In this case, it is best to use the folder `/tmp/`.

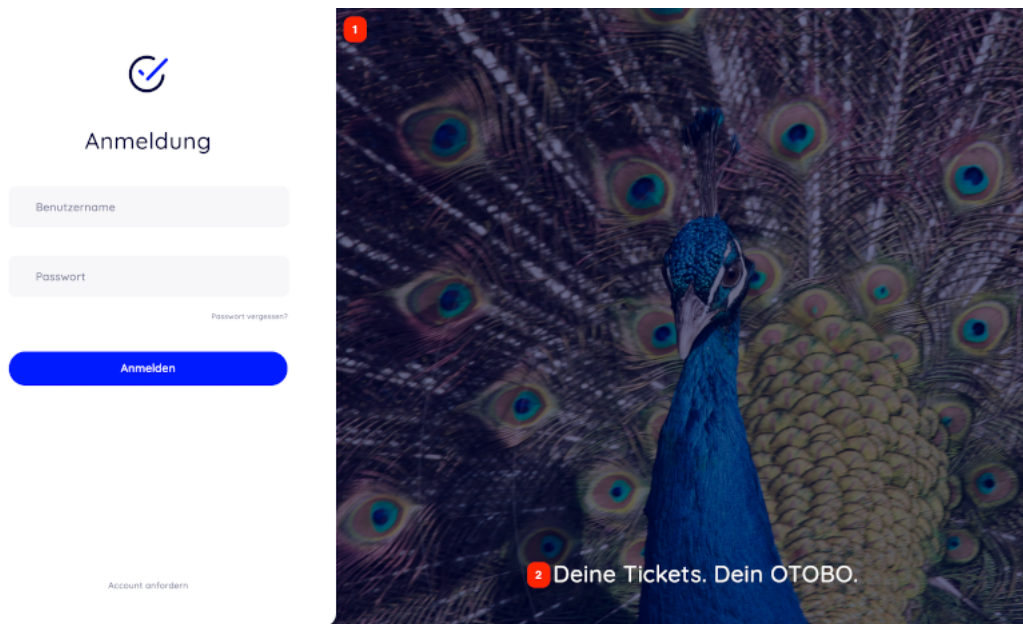
In the next step copy the Logo into the OTOBO Home directory:

```
**# Using OTOBO Docker Installation**
otobo_admin> docker cp /tmp/Logos.png otopo_web_1:/opt/otobo/var/httpd/htdocs/skins/
↳Customer/default/img/

**# Nativ installation in /opt/otobo/**
otobo_admin> cp /tmp/Logos.png /opt/otobo/var/httpd/htdocs/skins/Customer/default/img/
```

Now change inside the OTOBO Agent Interface to Admin -> System Configuration and change the following settings:

### 11.2.1 Change Customer Login Pictures and Text



- 1 and 2 - System Configuration Option **CustomerLogin::Settings**

#### Remove Opacity and Watermark

At the moment it is not possible to remove the overlay and watermark used in the image on the right by system configuration.

To remove the opacity, please adjust the option **#oooLoginBG > .oooBG** in the file `var/httpd/htdocs/skins/Customer/default/css/Core.Login.css`

```
#oooLoginBG > .oooBG {
    position: relative;
    width: 100%;
    height: 100%;
    /* opacity: 0.45; Disable opacity */
    background-size: cover;
    overflow: hidden;
}
```

To remove the watermark, please remove the following line inside the file:

`Kernel/Output/HTML/Templates/Standard/CustomerLogin.tt`

```
<!-- start login -->
<div id="oooLoginBG">
  <div class="oooBG" style="background-image: url([% Data.Background | html %]);">
# remove this line ->      <div id="oooBGSignet" style="background-image: url([%_
↳Config("Frontend::WebPath") %]common/img/otobo-signet_border.svg);"></div>
  </div>
  <h1>[% Translate(Data.LoginText) | html %]</h1>
</div>
```

---

**Nota:** Please add the files to a opm package in the next step, so that the changes remain persistent. You can find instructions on how to do this in our Admin Manual: <https://doc.otobo.org/manual/developer/10.1/en/content/how-to-publish-otobo-extensions.html>

---

## 11.2.2 Change Customer Dashboard tiles and options

To change the colours on the Customer Dashboard, please go to Admin -> System Configuration and search for **CustomerDashboard**.

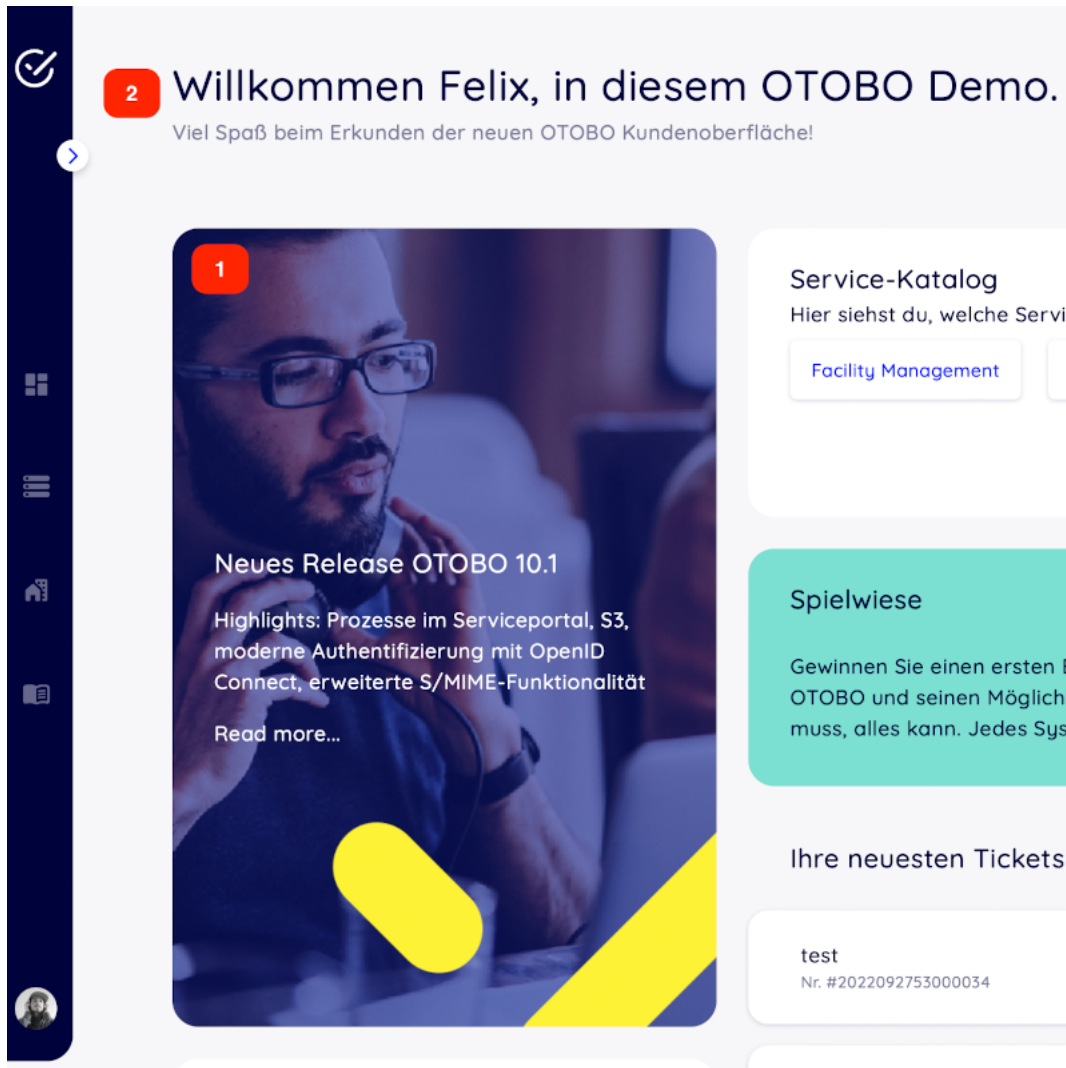
In the search result you will find all the options you need with colour definitions.

- 1 - To change the picture, link and text please use the System Configuration Option **CustomerDashboard::Tiles###FeaturedLink-01**
- 2 - To change the toplevel text please use the System Configuration Option **CustomerDashboard::Configuration::Text**

---

**Nota:** Please disable the config options from not needed tiles.

---



---

## Installing Perl Modules from CPAN

---

When there are special requirements then the need for additional Perl modules may arise. Fortunately, Perl has an excellent package repository that can satisfy almost all needs. That repository is called CPAN and is available at <https://metacpan.org/>.

It is recommended to use the command line client `cpanm` for installing modules. `cpanm` is often already installed on your system. Please see <https://metacpan.org/pod/App::cpanminus> for what to do when it isn't already available.

Alternatively, many Perl modules are also available as packages for your operating system. These packages can be installed with your system's regular package manager.

Per default `cpanm` installs modules into a systemwide location. In this case modules must be installed as the root user. For example, the command

```
root> cpanm Acme::Dice
```

resulta em:

```
otobo> perldoc -l Acme::Dice  
/usr/local/share/perl/5.30.0/Acme/Dice.pm
```

### 12.1 Docker-based installations

Special care must be taken when OTOBO runs under Docker. In this case an installation into a systemwide location would initially work as well. However, due to how Docker works, this installed modules would be lost when the container is restarted. Therefore the modules must be installed into a location that does survive a restart. The directory `/opt/otobo/local` within the volume **otobo\_opt\_otobo** can be used for that. Modules that are installed in `/opt/otobo/local` will be picked up by Perl because the environment variables `PERL5LIB` and `PATH` are preset accordingly.

The installed Perl modules will also be available after an upgrade of OTOBO. There is the general rule that files added to `/opt/otobo` won't be removed by an upgrade.

For installing Perl modules in a specific location we need to modify our install command. Specifically, we need to add the option `--local-lib`. Here is a sample session in the container **web**.

```
# starting a bash session in the container web
docker_admin> cd /opt/otobo-docker/
docker_admin> docker-compose exec web bash
otobo@6ef90ed00cd0:~$ pwd
/opt/otobo

# installing the sample module Acme::Dice
otobo@6ef90ed00cd0:~$ cpanm --local-lib local Acme::Dice
--> Working on Acme::Dice
Fetching http://www.cpan.org/authors/id/B/BO/BOFTX/Acme-Dice-1.01.tar.gz ... OK
Configuring Acme-Dice-1.01 ... OK
Building and testing Acme-Dice-1.01 ... OK
Successfully installed Acme-Dice-1.01
1 distribution installed

# confirm the installation directory
otobo@6ef90ed00cd0:~$ perldoc -l Acme::Dice
/opt/otobo/local/lib/perl5/Acme/Dice.pm

# locally installed module is found because the environment is preset accordingly
otobo@6ef90ed00cd0:~$ echo $PERL5LIB
/opt/otobo_install/local/lib/perl5:/opt/otobo/local/lib/perl5
otobo@6ef90ed00cd0:~$ echo $PATH
/opt/otobo_install/local/bin:/opt/otobo/local/bin:/usr/local/sbin:/usr/local/bin:/usr/
↪sbin:/usr/bin:/sbin:/bin
```

---

## Ajuste de desempenho

---

Esta é uma lista de técnicas de aprimoramento de desempenho para sua instalação OTOBO. Os tópicos incluem configuração, codificação, uso de memória e muito mais.

### 13.1 Módulo de Índice do Ticket

O módulo de índice de tickets pode ser definido através da configuração do sistema `Ticket::IndexModule`. Existem dois módulos de backend que constroem o índice para a Visão de filas:

**Kernel::System::Ticket::IndexAccelerator::RuntimeDB** Esta é a opção padrão, que gerará cada exibição de fila em tempo real a partir da tabela de tickets. Você não terá problemas de desempenho até ter cerca de 60.000 tickets abertos em seu sistema.

**Kernel::System::Ticket::IndexAccelerator::StaticDB** O módulo mais poderoso, deve ser usado quando você tiver acima de 80.000 tickets abertos. Ele usa uma tabela `ticket_index` extra, que será preenchida com palavras-chave com base nos dados do ticket. Use o seguinte comando para gerar um índice inicial após a alterar de back-ends:

```
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Ticket::QueueIndexRebuild
```

### 13.2 Índice de pesquisa de tickets

O OTOBO usa um índice de pesquisa especial para realizar pesquisas de texto completo nos campos de artigos de diferentes canais de comunicação.

Para criar um índice inicial, use este comando:

```
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Ticket::FulltextIndex --rebuild
```

**Nota:** A indexação real do artigo ocorre por meio de um trabalho do daemon do OTOBO em segundo plano. Enquanto os artigos que acabaram de ser adicionados ao sistema são marcados para indexação imediatamente, pode acontecer que o índice esteja disponível apenas após alguns minutos.

Existem algumas opções disponíveis para ajustar o índice de pesquisa:

**Ticket::SearchIndex::IndexArchivedTickets** Defina se os tickets arquivados serão incluídos no índice de pesquisa (desativado por padrão). É aconselhável manter o índice pequeno em sistemas grandes com tickets arquivados. Se isso estiver ativado, os tickets arquivados serão encontrados em pesquisas de texto completo.

**Ticket::SearchIndex::Attribute** Configurações básicas do índice de texto completo.



Fig. 13.1: Configuração Ticket::SearchIndex::Attribute

**Nota:** Execute o seguinte comando para gerar um novo índice:

```
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Ticket::FulltextIndexRebuild
```

**WordCountMax** Define o número máximo de palavras que serão processadas para criar o índice. Por exemplo, apenas as primeiras 1000 palavras de um corpo de artigo são armazenadas no índice de pesquisa de artigos.

**WordLengthMin e WordLengthMax** Usado como limites de comprimento de palavra. Somente palavras com um comprimento entre esses dois valores são armazenadas no índice de pesquisa de artigos.

**Ticket::SearchIndex::Filters** Filtros baseados em expressões regulares excluem partes do texto original do índice de texto completo.

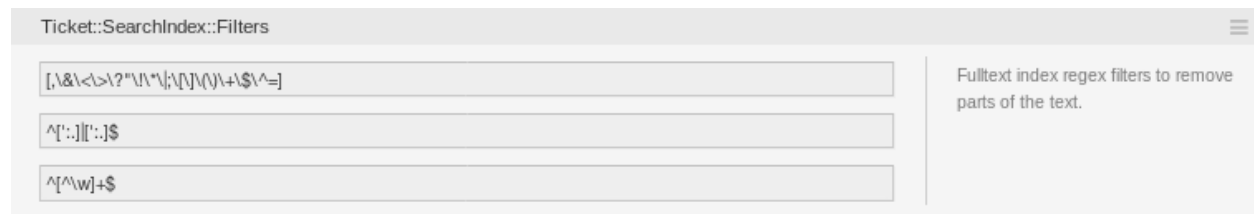


Fig. 13.2: Configuração Ticket::SearchIndex::Filters

Existem três filtros padrão definidos:

- O primeiro filtro remove caracteres especiais como: , & < > ? " ! \* | ; [ ] ( ) + \$ ^ =
- O segundo filtro remove as palavras que começam ou terminam com um dos seguintes caracteres: ' : .



- O terceiro filtro remove as palavras que não contêm um caractere de palavra: a-z, A-Z, 0-9,

–

**Ticket::SearchIndex::StopWords** Palavras de parada em inglês para o índice de texto completo. Essas palavras serão removidas do índice de pesquisa.



Fig. 13.3: Configuração `Ticket::SearchIndex::StopWords###en`

Existem chamadas stop-words definidas para alguns idiomas. Essas stop-words serão ignoradas ao criar o índice de pesquisa.

**Ver também:**

Se seu idioma não estiver nas definições de configuração do sistema ou você desejar adicionar mais palavras, você poderá adicioná-las a esta configuração:

- `Ticket::SearchIndex::StopWords###Custom`

## 13.3 Pesquisa de documentos

O OTOBO usa o Elasticsearch para sua funcionalidade de pesquisa de documentos. Para uma boa introdução aos conceitos, instalação e uso do Elasticsearch, siga as instruções em [Getting Started guide](#).

### 13.3.1 Tamanho Heap

O Elasticsearch é escrito em Java e, portanto, é executado em uma Java Virtual Machine (JVM) em qualquer nó do cluster. Essa JVM usa uma parte da memória, chamada heap, cujo tamanho pode ser configurado no arquivo de configuração `jvm.options`.

As configurações mínima e máxima do heap são definidas por padrão para um valor de 1 GB e podem ser modificadas com as seguintes opções:

- Tamanho mínimo de heap `Xms1g`.
- Tamanho máximo de heap `Xmx1g`.

Se o `Xms` tiver um valor menor que o `Xmx`, a JVM redimensionará o heap usado sempre que o limite atual for excedido, até que o valor de `Xmx` seja atingido. Esse redimensionamento faz com que o serviço seja

pausado até ser concluído, o que pode diminuir a velocidade e a reatividade das ações de pesquisa ou indexação. Portanto, é altamente recomendável definir essas configurações para um valor igual.

**Aviso:** Se o tamanho máximo do heap for excedido, o nó do cluster relacionado para de funcionar e pode até desligar o serviço.

Quanto maior o valor máximo do heap for definido, mais memória poderá ser usada pelo Elasticsearch, o que também aumentará as possíveis pausas para coleta de lixo, realizadas pela JVM. Portanto, é recomendável definir um valor para `xmx`, que não exceda 50% da memória física.

Para obter mais informações e boas regras práticas sobre o tamanho da pilha, acesse a [official documentation <https://www.elastic.co/guide/en/elasticsearch/reference/current/heap-size.html>](https://www.elastic.co/guide/en/elasticsearch/reference/current/heap-size.html).

### 13.3.2 Alocação de disco

Ao executar o serviço, o Elasticsearch inspeciona o espaço em disco disponível. Com base no resultado, ele decide se alocar novos shards para um nó do cluster. Em alguns casos, ele até realoca os fragmentos longe de um nó. Esse comportamento é determinado pela capacidade do disco atual. Ele pode ser definido pelas configurações no arquivo de configuração `elasticsearch.yml`. Aqui estão algumas configurações relevantes. Eles vêm com bons valores padrão, mas podem ser importantes na solução de problemas.

**`cluster.routing.allocation.disk.watermark.low`** Valor padrão de 85%. Quando esse limite for excedido, o Elasticsearch não alocará mais shards para o nó do cluster relacionado. A operação desse nó não é influenciada e os dados ainda podem ser indexados e pesquisados.

**`cluster.routing.allocation.disk.watermark.high`** Valor padrão de 90%. Quando esse limite for excedido, o Elasticsearch tentará realocar os shards existentes para outros nós que tenham espaço suficiente disponível.

**`cluster.routing.allocation.disk.watermark.flood_stage`**

Valor padrão de 95%. Quando esse limite é excedido, o Elasticsearch atualiza a configuração de todos os índices, que têm pelo menos um fragmento alocado para o nó do cluster relacionado, para blocos de índice somente leitura. Especificamente, eles são sinalizados com `index.blocks.read_only_allow_delete`.

Após essa atualização, não é mais possível indexar novos dados a esses índices. Os índices são restritos a pesquisas e para excluir ações apenas.

---

**Nota:** Se o flood stage for excedido e determinados índices estiverem configurados no modo somente leitura, essa configuração não será alterada automaticamente pelo Elasticsearch. Se os discos relacionados contiverem espaço livre suficiente novamente devido a ações manuais, será necessário alterar a configuração novamente para o modo normal manualmente.

---

Para obter mais informações sobre marcas d'água de disco e alocação de shard com base em disco, acesse [the official documentation](#).

## 13.4 Armazenamento de artigos

Existem dois módulos de back-end diferentes para o armazenamento de artigos por telefone, email e artigos internos. O armazenamento de artigos usado pode ser configurado na configuração `Ticket::Article::Backend::MIMEBase::ArticleStorage`.

**Kernel::System::Ticket::Article::Backend::MIMEBase::ArticleStorageDB** Este módulo padrão armazenará anexos no banco de dados. Também funciona com vários servidores front-end, mas requer muito espaço de armazenamento no banco de dados.

---

**Nota:** Não use isso com instalações grandes.

---

**Kernel::System::Ticket::Article::Backend::MIMEBase::ArticleStorageFS** Use este módulo para armazenar anexos no sistema de arquivos local. É rápido, mas se você tiver vários servidores front-end, verifique se o sistema de arquivos é compartilhado entre os servidores. Coloque-o em um compartilhamento NFS ou, de preferência, em uma SAN ou solução semelhante.

---

**Nota:** Recomendado para grandes instalações.

---

Você pode alternar de um back-end para outro em tempo real. Você pode alternar o back-end na configuração do sistema e, em seguida, executar este utilitário de linha de comando para colocar os artigos do banco de dados no sistema de arquivos ou vice-versa:

```
otobo> /opt/otobo/bin/otobo.Console.pl Admin::Article::StorageSwitch --target 
↔ArticleStorageFS
```

Você pode usar o `--target` opção para especificar o backend de destino.

---

**Nota:** Todo o processo pode levar um tempo considerável para ser executado, dependendo do número de artigos que você possui e da energia da CPU disponível e / ou capacidade da rede.

---

Se você deseja manter anexos antigos no banco de dados, pode ativar a opção de configuração do sistema `Ticket::Article::Backend::MIMEBase::CheckAllStorageBackends` para garantir que o OTOBO ainda os encontre.

## 13.5 Arquivando Tickets

Como o OTOBO pode ser usado como um sistema à prova de auditoria, excluir tickets fechados pode não ser uma boa ideia. Portanto, há um recurso que permite arquivar tickets.

Os tickets que correspondem a determinados critérios podem ser marcados como arquivados. Esses tickets não serão acessados se você fizer uma pesquisa regular de tickets ou executar um trabalho de agente genérico. O sistema em si não precisa mais lidar com uma quantidade enorme de tickets, pois apenas os tickets mais recentes são levados em consideração ao usar o OTOBO. Isso pode resultar em um enorme ganho de desempenho em sistemas grandes.

Para usar o recurso de arquivamento:

1. Ative a configuração `Ticket::ArchiveSystem` nas Configurações de Sistema.

### 2. Defina uma tarefa do agente genérico:

- Clique no botão Adicionar Tarefa na tela Atendente genérico.
- Configurações da tarefa forneça um nome para o trabalho de arquivamento.
- Execução automática: selecione as opções apropriadas para agendar essa tarefa.
- Selecionar chamados: Pode ser uma boa ideia arquivar apenas os tickets em um estado fechado que foram fechados há alguns meses antes.
- Alterar/Adicionar Atributos do Chamado: defina o campo\*Arquivar chamados selecionados\* to Arquivar os chamados.
- Salve a tarefa no final da página.
- Clique no link \* Executar esta Tarefa \* na tabela de visão geral para ver os tickets afetados.
- Clique no botão Executar Tarefa.

---

**Nota:** Até 5000 tickets podem ser modificados executando esta tarefa manualmente.

---

Ao procurar tickets, o padrão do sistema é procurar tickets que não estão arquivados.

Para procurar tickets arquivados:

1. Abra a tela de pesquisa de tickets.
2. Defina Procurar Arquivo como Tickets não arquivados ou \* Todos os tickets \*.
3. Realize a pesquisa.

## 13.6 Armazenamento em cache

Um módulo de cache rápido é uma grande ajuda em termos de desempenho. Recomendamos usar um servidor Redis Cache ou criar um ramdisk.

### 13.6.1 Instale um servidor de cache Redis

1. Instale servidor Redis

Antes de tudo, você precisa instalar o mais novo servidor Redis. A maneira mais fácil é [setup Redis](#) no mesmo host que OTOBO esta instalado à sua porta padrão.

2. Instale o módulo Perl Redis ou Redis::Fast

Você pode escolher qual módulo Redis usar: Redis ou 'Redis::Fast' (que é compatível com Redis mas **~2x faster**). Por favor use nosso `otobo.CheckModules.pl --list` para escolher o pacote certo para você:

```
otobo> /opt/otobo/bin/otobo.CheckModules.pl --all
```

3. Configurar OTOBO para Redis

Por favor, use o OTOBO SysConfig (Administração-> Configuração do sistema) para configurar o OTOBO corretamente:

Setting	Description	Default value
Cache::Redis###Server	Redis server URL	127.0.0.1:6379
Cache::Redis###DatabaseNumber	Number of logical database	0
Cache::Redis###RedisFast	Use or not Redis::Fast	0
Cache::Module	Activate Redis Cache Module	DB (use Redis)

### 13.6.2 Cache do RamDisk

O OTOBO armazena muitos dados de cache temporários no `/opt/otobo/var/tmp`. Verifique se ele usa um sistema de arquivos e armazenamento de alto desempenho. Se você possui RAM suficiente, também pode tentar colocar este diretório em um ramdisk como este:

```
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Session::DeleteAll
otobo> /opt/otobo/bin/otobo.Console.pl Maint::Cache::Delete
root> mount -o size=16G -t tmpfs none /opt/otobo/var/tmp
```

**Nota:** Adicione ponto de montagem persistente no `/etc/fstab`.

**Aviso:** Este será um armazenamento não permanente que será perdido na reinicialização do servidor. Todas as suas sessões (se você as armazenar no sistema de arquivos) e os dados do cache serão perdidos.

## 13.7 Clustering

Para cargas muito altas, pode ser necessário operar o OTOBO em um cluster de vários servidores front-end. Esta é uma tarefa complexa com muitas armadilhas. É altamente recomendável entrar em contato com nossos especialistas antes de tentar implementar isso por conta própria.



---

### Histórico da documentação

---

1. 2019 - OTRS Guia de instalação - OTRS AG (<https://otrs.com>)
2. 2020 - OTOBO Manual de instalação - Rother OSS GmbH (<https://otobo.de>)

Publicado por: Rother OSS GmbH, (<https://otobo.de>), Oberwaling 31, 94339 Leiblfing, Germany.  
Authors: OTRS AG (original version), Rother OSS GmbH (<https://otobo.de>)

É concedida permissão para copiar, distribuir e/ou modificar este documento sob os termos da GNU Free Documentation License, Versão 1.3 ou qualquer versão posterior publicada pela Free Software Foundation; sem secções Invariantes, sem textos de capa e sem textos de contracapa. Uma cópia da licença pode ser encontrada no site GNU <<https://www.gnu.org/licenses/fdl-1.3.txt>> \_\_.