Zammad

Jan 18, 2019

About

The Zammad documentation consists of three parts:

- Zammad system installation and configuration (this documentation)
- Zammad administration (https://admin-docs.zammad.org)
- Zammad user documentation (https://user-docs.zammad.org)

This system documentation for Zammad is organized into a couple of sections:

CHAPTER 1

Zammad

Do you receive many emails and want to answer them with a team of agents?

You're going to love Zammad!

Zammad is a web based open source helpdesk/customer support system with many features to manage customer communication via several channels like telephone, facebook, twitter, chat and emails. It is distributed under version 3 of the GNU AFFERO General Public License (GNU AGPLv3).

The code is open source, and available on GitHub!

CHAPTER 2

Software

If you want to install Zammad, you need the following software.

2.1 1. Ruby Programming Language

Zammad requires Ruby. All required rubygems like ruby on rails are listed in the Gemfile. The following Ruby version is supported:

- Ruby 2.4.4

2.2 2. Database Server

Zammad will store all content in an RDBMS. You can choose between the following products:

- MySQL 5.6+
- MariaDB 10.0+
- PostgreSQL 9.1+

Note: We tend to recommend PostgreSQL. For the last 10 years we had the best experience with it.

Warning: Required configuration for MySQL/MariaDB:

- Use UTF8 encoding. utf8mb4 for example will fail.
- Set `max_allowed_packet` to a value larger than the default of 4 MB (64 MB+ recommended).

2.3 3. Reverse Proxy

In a typical web environment today, you use a reverse proxy to deliver the static content of your application. Only the "expensive" app required HTTP requests are forwarded to the application server.

The following reverse proxies are supported:

- Nginx 1.3+
- Apache 2.2+

2.4 4. Elasticsearch

For excellent search performance we use Elasticsearch. The following Elasticsearch versions are supported:

- Elasticsearch 2.4 up to 5.5 with `mapper-attachments` plugin
- Elasticsearch 5.6 with `ingest-attachment` plugin

CHAPTER 3

Hardware

You can run Zammad on bare metal or on a virtual machine. Choose what you prefer.

3.1 For Zammad and a database server like PostgreSQL we recommend at least:

- 2 CPU cores
- 2 GB of RAM (+2 GB if you want to run Elasticsearch on the same server)

3.2 For optimal performance up to 40 agents:

- 4 CPU cores
- 4 GB of RAM (+4 GB if you want to run Elasticsearch on the same server)

Of course at the end it depends on acutal load of concurent agents and data traffic.

Note: We can't suggest any disk space recommendations, as this highly depends on how you work. Zammad will always try to recognize the same attachments and store it just once.

CHAPTER 4

Install from source (generic)

4.1 Setup Elasticsearch

Elasticsearch is a dependency of Zammad and needs to be provided before installing Zammad. Please take a look at the following page: *Set up Elasticsearch* .

4.1.1 1. Install Zammad on your system

You can directly download Zammad from https://ftp.zammad.com/ or use the direct URL to get the latest stable release via https://ftp.zammad.com/zammad-latest.tar.gz

```
root@shell> useradd zammad -m -d /opt/zammad -s /bin/bash
root@shell> cd /opt
root@shell> wget https://ftp.zammad.com/zammad-latest.tar.gz
root@shell> tar -xzf zammad-latest.tar.gz -C zammad
root@shell> su - zammad
```

4.1.2 2. Install all dependencies

Please note that a working ruby 2.4.4 environment is needed.

```
zammad@shell> gem install bundler rake rails
```

4.2 For PostgreSQL (note, the option says “without . . . mysql”)

```
zammad@shell> bundle install --without test development mysql
```

4.3 For MySQL (note, the option says “without . . . postgres”)

```
zammad@shell> bundle install --without test development postgres
```

4.3.1 3. Configure your databases

```
zammad@shell> cp config/database/database.yml config/database.yml
zammad@shell> vi config/database.yml
```

4.3.2 4. Initialize your database

```
zammad@shell> export RAILS_ENV=production
zammad@shell> export RAILS_SERVE_STATIC_FILES=true
zammad@shell> rake db:create
zammad@shell> rake db:migrate
zammad@shell> rake db:seed
```

4.3.3 5. Change directory to zammad (if needed) and start services:

```
zammad@shell> rake assets:precompile
```

You can start all services by hand or use systemd to start / stop Zammad.

4.4 Starting all servers manually

```
zammad@shell> rails s -p 3000 # application web server
zammad@shell> script/websocket-server.rb start # non blocking websocket server
zammad@shell> script/scheduler.rb start # generate overviews on demand, just send␣
↪changed data to browser
```

4.5 Starting servers with Systemd

```
zammad@shell> cd scripts/systemd
zammad@shell> sudo ./install-zammad-systemd-services.sh
```

4.5.1 6. Go to http://localhost:3000 and you’ll see:

- “Welcome to Zammad!”, there you need to create your admin user and invite other agents.

4.6 Reset a Zammad installation (for a fresh start after testing)

Please note: this actions will delete all existing data! Dont use it on a production system.

```
zammad@shell>sudo systemctl stop zammad
zammad@shell>rake db:drop
zammad@shell>rake db:create
zammad@shell>rake db:migrate
zammad@shell>rake db:seed
zammad@shell>sudo systemctl start zammad
```

CHAPTER 5

Install from source (Debian 7, 8 / Ubuntu 16.04 / Ubuntu 18.04)

5.1 Prerequisites

```
apt-get update
apt-get install curl git-core patch build-essential bison zlib1g-dev libssl-dev␣
↪libxml2-dev libxml2-dev sqlite3 libsqlite3-dev autotools-dev libxslt1-dev libyaml-0-
↪2 autoconf automake libreadline6-dev libyaml-dev libtool libgmp-dev libgdbm-dev␣
↪libncurses5-dev pkg-config libffi-dev libmysqlclient-dev mysql-server nginx gawk
```

5.2 Add User

```
useradd zammad -m -d /opt/zammad -s /bin/bash
echo "export RAILS_ENV=production" >> /opt/zammad/.bashrc
```

5.3 Create MySQL user zammad (for Debian: upgrade MySQL to v5.6+ before, see: http://dev.mysql.com/downloads/repo/apt/)

```
mysql --defaults-extra-file=/etc/mysql/debian.cnf -e "CREATE USER 'zammad'@'localhost
↪' IDENTIFIED BY 'Your_Pass_Word'; GRANT ALL PRIVILEGES ON zammad_prod.* TO 'zammad'@
↪'localhost'; FLUSH PRIVILEGES;"
```

5.4 Get Zammad

```
su zammad
cd ~
curl -O https://ftp.zammad.com/zammad-latest.tar.gz
tar -xzf zammad-latest.tar.gz
rm zammad-latest.tar.gz
```

5.5 Install environnment

```
gpg --keyserver hkp://keys.gnupg.net --recv-keys␣
↪409B6B1796C275462A1703113804BB82D39DC0E3
curl -L https://get.rvm.io | bash -s stable
source /opt/zammad/.rvm/scripts/rvm
echo "source /opt/zammad/.rvm/scripts/rvm" >> /opt/zammad/.bashrc
echo "rvm --default use 2.4.4" >> /opt/zammad/.bashrc
rvm install 2.4.4
gem install bundler
```

5.6 Install Zammad

```
bundle install --without test development postgres
cp config/database/database.yml config/database.yml
```

- insert mysql user, pass & change adapter to mysql2 & change database to zammad_prod

```
vi config/database.yml
```

```
rake db:create
rake db:migrate
rake db:seed
rake assets:precompile
```

5.7 Start Zammad

```
rails s -p 3000 &>> log/zammad.log &
script/websocket-server.rb start &>> log/zammad.log &
script/scheduler.rb start &>> log/zammad.log &
```

5.8 Create Nginx Config & restart Nginx

```
exit
cp /opt/zammad/contrib/nginx/zammad.conf /etc/nginx/sites-available/zammad.conf
```

- change servername “localhost” to your domain if your’re not testing localy

```
vi /etc/nginx/sites-available/zammad.conf
```

```
ln -s /etc/nginx/sites-available/zammad.conf /etc/nginx/sites-enabled/zammad.
↪conf
```

systemctl restart nginx

5.9 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

CHAPTER 6

Install from source (Mac OS 10.8)

- Install Xcode from the App Store, open it -> Xcode menu > Preferences > Downloads -> install command line tools

```
curl -L https://get.rvm.io | bash -s stable --ruby
source ~/.rvm/scripts/rvm
start new shell -> ruby -v
```

```
test -d ~/zammad/ || mkdir ~/zammad
cd ~/zammad/
curl -L -O https://ftp.zammad.com/zammad-latest.tar.bz2 | tar -xj
```

```
cd zammad-latest
bundle install
sudo ln -s /usr/local/mysql/lib/libmysqlclient.18.dylib /usr/lib/libmysqlclient.18.
↪dylib # if needed!
rake db:create
rake db:migrate
rake db:seed
```

```
cd zammad-latest
cp config/database/database.yml config/database.yml
rake db:create
rake db:migrate
rake db:seed
```

```
puma -p 3000 # application web server
script/websocket-server.rb start # non blocking websocket server
script/scheduler.rb start # generate overviews on demand, just send changed data to␣
↪browser
```

- http://localhost:3000/#getting_started

CHAPTER 7

Install on CentOS via RPM

Note: Currently we support RHEL7 & CentOS7.

7.1 Prerequisites

7.1.1 Setup Elasticsearch

Elasticsearch is a dependency of Zammad and needs to be provided before installing Zammad. Please take a look at the following page: *Set up Elasticsearch* .

7.2 Add Zammad-Repo and Install Zammad

```
sudo yum -y install epel-release wget
sudo wget -O /etc/yum.repos.d/zammad.repo https://dl.packager.io/srv/zammad/zammad/
↪stable/installer/el/7.repo

sudo yum -y install zammad
```

7.2.1 SeLinux & Firewalld

On Centos SeLinux & Firewalld are possibly enabled. To get everything work you have to use the following commands:

```
setsebool httpd_can_network_connect on -P
firewall-cmd --zone=public --add-service=http --permanent
firewall-cmd --zone=public --add-service=https --permanent
firewall-cmd --reload
```

7.3 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

7.3.1 Change your webserver configuration (non localhost connections):

Add your fully qualified domain name or public IP to server name directive in your web server configuration and restart your web server. The installer will give you a hint where Zammad's web server config file is located.

7.3.2 nginx

/etc/nginx/conf.d/zammad.conf

```
server {
    listen 80;

    # replace 'localhost' with your fqdn if you want to use zammad from remote
    server_name localhost;
```

7.4 You can manage the Zammad services manually:

7.4.1 Zammad

```
sudo systemctl status zammad
sudo systemctl stop zammad
sudo systemctl start zammad
sudo systemctl restart zammad
```

7.4.2 Only web application server

```
sudo systemctl status zammad-web
sudo systemctl stop zammad-web
sudo systemctl start zammad-web
sudo systemctl restart zammad-web
```

7.4.3 Only worker process

```
sudo systemctl status zammad-worker
sudo systemctl stop zammad-worker
sudo systemctl zammad-worker
sudo systemctl restart zammad-worker
```

7.4.4 Only websocket server

```
sudo systemctl status zammad-websocket
sudo systemctl stop zammad-websocket
sudo systemctl start zammad-websocket
sudo systemctl restart zammad-websocket
```

CHAPTER 8

Install on Debian via DEB

Note: Currently we support Debian 8 and 9

8.1 Prerequisites

Be sure to use an UTF-8 locale or PostgreSQL will not install.

8.1.1 Setup Elasticsearch

Elasticsearch is a dependency of Zammad and needs to be provided before installing Zammad. Please take a look at the following page: *Set up Elasticsearch* .

8.1.2 Check locale

```
locale
```

If there is nothing with UTF-8 in the name shown like “LANG=en_US.UTF-8” you have to set a new locale.

8.1.3 Set locale

```
sudo apt-get install apt-transport-https locales sudo wget
sudo locale-gen en_US.UTF-8
sudo echo "LANG=en_US.UTF-8" > /etc/default/locale
```

8.2 Add Zammad DEB repo and install

```
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
```

8.2.1 For Debian 8

```
sudo wget -O /etc/apt/sources.list.d/zammad.list https://dl.packager.io/srv/zammad/
↪zammad/stable/installer/debian/8.repo
```

8.2.2 For Debian 9

```
sudo wget -O /etc/apt/sources.list.d/zammad.list https://dl.packager.io/srv/zammad/
↪zammad/stable/installer/debian/9.repo
```

```
sudo apt-get update
sudo apt-get install zammad
```

8.3 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

8.4 Change your webserver configuration (non localhost connections):

Add your fully qualified domain name or public IP to server name directive in your web server configuration and restart your web server. The installer will give you a hint where Zammad's web server config file is located.

8.4.1 nginx

/etc/nginx/sites-enabled/zammad.conf

```
server {
    listen 80;

    # replace 'localhost' with your fqdn if you want to use zammad from remote
    server_name localhost;
```

8.5 You can manage the Zammad services manually:

8.5.1 Zammad

```
sudo systemctl status zammad
sudo systemctl stop zammad
sudo systemctl start zammad
sudo systemctl restart zammad
```

8.5.2 only web application server

```
sudo systemctl status zammad-web
sudo systemctl stop zammad-web
sudo systemctl start zammad-web
sudo systemctl restart zammad-web
```

8.5.3 only worker process

```
sudo systemctl status zammad-worker
sudo systemctl stop zammad-worker
sudo systemctl start zammad-worker
sudo systemctl restart zammad-worker
```

8.5.4 only websocket server

```
sudo systemctl status zammad-websocket
sudo systemctl stop zammad-websocket
sudo systemctl start zammad-websocket
sudo systemctl restart zammad-websocket
```

CHAPTER 9

Install on Ubuntu via DEB

Note: We currently support Ubuntu 16.04 LTS and 18.04 LTS.

9.1 Prerequisites

Be sure to use an UTF-8 locale or PostgreSQL will not install.

9.1.1 Setup Elasticsearch

Elasticsearch is a dependency of Zammad and needs to be provided before installing Zammad. Please take a look at the following page: *Set up Elasticsearch* .

9.1.2 Check locale

```
locale
```

If there is nothing with UTF-8 in the name shown like “LANG=en_US.UTF-8” you have to set a new locale.

9.1.3 Set locale

```
apt-get install apt-transport-https locales sudo wget
locale-gen en_US.UTF-8
echo "LANG=en_US.UTF-8" > /etc/default/locale
```

9.2 Add Zammad DEB Repo

9.2.1 Ubuntu 16.04

```
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list \
  https://dl.packager.io/srv/zammad/zammad/stable/installer/ubuntu/16.04.repo
```

9.2.2 Ubuntu 18.04

```
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list \
  https://dl.packager.io/srv/zammad/zammad/stable/installer/ubuntu/18.04.repo
```

9.2.3 Install on Ubuntu (16.04 and 18.04)

```
sudo apt-get update
sudo apt-get install zammad
```

Note: You might need to apt-get install wget apt-transport-https for the above instructions to work.

9.3 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

9.4 Change your webserver configuration (non localhost connections):

Add your fully qualified domain name or public IP to server name directive in your web server configuration and restart your web server. The installer will give you a hint where Zammad's web server config file is located.

9.4.1 nginx

/etc/nginx/sites-enabled/zammad.conf

```
server {
    listen 80;

    # replace 'localhost' with your fqdn if you want to use zammad from remote
    server_name localhost;
```

9.5 You can manage the Zammad services manually:

9.5.1 Zammad

```
sudo systemctl status zammad
sudo systemctl stop zammad
sudo systemctl start zammad
sudo systemctl restart zammad
```

9.5.2 Only web application server

```
sudo systemctl status zammad-web
sudo systemctl stop zammad-web
sudo systemctl start zammad-web
sudo systemctl restart zammad-web
```

9.5.3 Only worker process

```
sudo systemctl status zammad-worker
sudo systemctl stop zammad-worker
sudo systemctl start zammad-worker
sudo systemctl restart zammad-worker
```

9.5.4 Only websocket server

```
sudo systemctl status zammad-websocket
sudo systemctl stop zammad-websocket
sudo systemctl start zammad-websocket
sudo systemctl restart zammad-websocket
```

CHAPTER 10

Install on SUSE via RPM

Note: Currently we support SLES 12 and OpenSUSE with versions 42.2 and 42.3

Warning: OpenSUSE LEAP 15.0 hasn't been tested yet, but should work as well.

10.1 Install dependencies

10.1.1 Setup Elasticsearch

Elasticsearch is a dependency of Zammad and needs to be provided before installing Zammad. Please take a look at the following page: *Set up Elasticsearch* .

10.1.2 nginx on SLES12

```
sudo zypper addrepo "http://nginx.org/packages/sles/12" "nginx"
```

10.2 Add Zammad RPM repo and install

```
sudo zypper install wget
sudo wget -O /etc/zypp/repos.d/zammad.repo https://dl.packager.io/srv/zammad/zammad/
↪stable/installer/sles/12.repo
sudo zypper install zammad
```

10.3 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

Note: Make sure that the firewall is not blocking port 80 (configure firewall via "yast firewall" or stop it via "systemctl stop SuSEfirewall2").

10.4 Change your webserver configuration (non localhost connections):

Add your fully qualified domain name or public IP to server name directive in your web server configuration and restart your web server. The installer will give you a hint where Zammad's web server config file is located.

10.4.1 nginx

/etc/nginx/sites-enabled/zammad.conf

```
server {
    listen 80;

    # replace 'localhost' with your fqdn if you want to use zammad from remote
    server_name localhost;
```

10.5 You can manage the Zammad services manually:

10.5.1 Zammad

```
sudo systemctl status zammad
sudo systemctl stop zammad
sudo systemctl start zammad
sudo systemctl restart zammad
```

10.5.2 Only web application server

```
sudo systemctl status zammad-web
sudo systemctl stop zammad-web
sudo systemctl start zammad-web
sudo systemctl restart zammad-web
```

10.5.3 Only worker process

```
sudo systemctl status zammad-worker
sudo systemctl stop zammad-worker
sudo systemctl zammad-worker
sudo systemctl restart zammad-worker
```

10.5.4 Only websocket server

```
sudo systemctl status zammad-websocket
sudo systemctl stop zammad-websocket
sudo systemctl start zammad-websocket
sudo systemctl restart zammad-websocket
```

CHAPTER 11

Set up Elasticsearch

We use Elasticsearch for the awesome search in Zammad.

Currently we support:

- Elasticsearch 2.4.x to 5.5.x with mapper-attachments plugin
- Elasticsearch 5.6.x with ingest-attachment plugin

This manual uses the “zammad” command which is only available if you installed Zammad from one of our package repos.

CHAPTER 12

Install Elasticsearch and its Attachment plugin

12.1 Generic install Elasticsearch 2.4 (mapper-attachments):

- Download and install via https://www.elastic.co/downloads/elasticsearch (2.4.x)
- Install the Attachment plugin

```
cd /usr/share/elasticsearch
bin/plugin install mapper-attachments
```

- Start elasticsearch

12.2 Generic install Elasticsearch 5.0-5.5 (mapper-attachments):

- Download and install via https://www.elastic.co/downloads/elasticsearch (5.0-5.5)
- Install the Attachment plugin

```
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install mapper-attachments
```

- Setting vm.max_map_count for Elasticsearch

```
sysctl -w vm.max_map_count=262144
```

- Start elasticsearch

12.3 Generic install Elasticsearch 5.6 (ingest-attachment):

- Download and install via https://www.elastic.co/downloads/elasticsearch (5.6)
- Install the Attachment plugin

```
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-attachment
```

- Setting vm.max_map_count for Elasticsearch

```
sysctl -w vm.max_map_count=262144
```

- Start elasticsearch

12.4 CentOS 7:

```
rpm --import https://artifacts.elastic.co/GPG-KEY-elasticsearch
echo "[elasticsearch-5.x]
name=Elasticsearch repository for 5.x packages
baseurl=https://artifacts.elastic.co/packages/5.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md"| sudo tee /etc/yum.repos.d/elasticsearch-5.x.repo
yum install -y java-1.8.0-openjdk elasticsearch
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-attachment
systemctl start elasticsearch
systemctl enable elasticsearch
```

12.5 Debian 8:

```
apt-get install apt-transport-https sudo wget
echo "deb http://ftp.debian.org/debian jessie-backports main" | sudo tee -a /etc/apt/
↪sources.list.d/debian-backports.list
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /
↪etc/apt/sources.list.d/elastic-5.x.list
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
apt-get update
apt-get install -t jessie-backports openjdk-8-jre
apt-get install elasticsearch
sudo /var/lib/dpkg/info/ca-certificates-java.postinst configure
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-attachment
systemctl restart elasticsearch
systemctl enable elasticsearch
```

12.6 Debian 9:

```
apt-get install apt-transport-https sudo wget
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /
↪etc/apt/sources.list.d/elastic-5.x.list
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
apt-get update
apt-get install openjdk-8-jre elasticsearch
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-attachment
```

(continues on next page)

(continued from previous page)

```
systemctl restart elasticsearch
systemctl enable elasticsearch
```

12.7 Ubuntu 16.04 & 18.04:

```
echo "deb https://artifacts.elastic.co/packages/5.x/apt stable main" | sudo tee -a /
↪etc/apt/sources.list.d/elastic-5.x.list
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
apt-get update
apt-get install openjdk-8-jre elasticsearch
sudo /usr/share/elasticsearch/bin/elasticsearch-plugin install ingest-attachment
systemctl restart elasticsearch
systemctl enable elasticsearch
```

CHAPTER 13

Configure Zammad to work with Elasticsearch

```
zammad run rails r "Setting.set('es_url', 'http://localhost:9200')"
```

CHAPTER 14

Create Elasticsearch index

After you have configured Zammad for using Elasticsearch, you need to rebuild the index with the following command:

```
zammad run rake searchindex:rebuild
```

CHAPTER 15

Optional settings

```
zammad run rails r "Setting.set('es_user', 'elasticsearch')"
zammad run rails r "Setting.set('es_password', 'zammad')"
```

```
zammad run rails r "Setting.set('es_index', Socket.gethostname + '_zammad')"
```

```
zammad run rails r "Setting.set('es_attachment_ignore', [ '.png', '.jpg', '.jpeg', '.
↪mpeg', '.mpg', '.mov', '.bin', '.exe', '.box', '.mbox' ] )"
```

```
zammad run rails r "Setting.set('es_attachment_max_size_in_mb', 50)"
```

CHAPTER 16

Using Elasticsearch on another server

Elasticsearch can also be installed on another server but you have to know that this is insecure out of the box because Elasticsearch has no authentication. For this reason you should run elasticsearch on 127.0.0.1 and use a reverse proxy with authentication to access it from Zammad.

You can find an Nginx reverse proxy config here:

- https://github.com/zammad/zammad/blob/develop/contrib/nginx/elasticsearch.conf

CHAPTER 17

List of values which are stored in ElasticSearch

17.1 Ticket

Please note that these fields may vary if you created custom fields (objects) in the admin interface.

Field	Sample Value	Description
article	Article	Article Hash, which includes all articles stored on a ticket
article_count	1	Count of articles
close_at	null	First close time, after create
close_diff_in_min	null	Business hours in minutes within or above the specified SLA for closing
close_escalation_at	null	Time stamp of the escalation if the SLA of the closing time has been vio
close_in_min	null	Business hours in minutes it took to close the ticket.
create_article_sender	Customer	Who has created the first article (Agent,Customer)
create_article_sender_id	2	Sender id of the first article (Agent|Customer)
create_article_type	web	Article type for the first article (note, email, phone. . .)
create_article_type_id	11	Article type ID for the first article (note, email, phone. . .)
created_at	2017-08-03T14:21:38.701Z	Created timestamp (DateTime, UTC)
created_by	User	User details of the user who created the ticket
created_by_id	13	User id of user who created the ticket
customer	User	Customer details
customer_id	13	User id of the current customer (assigned to ticket)
escalation_at	null	Next first escalation date (nearest close_escalation_at, first_response_es
first_response_at	null	Time stamp of the first reaction to the customer (DateTime, UTC)
first_response_diff_in_min	null	Business hours in minutes within or above the specified SLA for the firs
first_response_escalation_at	null	Time stamp of the escalation if the SLA of the first reaction time has be
first_response_in_min	null	Business hours in minutes it took to send inital response to customer.
group	Sales	Current ticket group (Sales, Support. . .)
group_id	1	Current ticket group id
id	19	Ticket id
last_contact_agent_at	null	Last contact to customer from agent, timestamp (DateTime, UTC)

Table 1 – continued from previous page

last_contact_at	2017-08-03T14:21:38.701Z	Last contact timestamp (DateTime, UTC)
last_contact_customer_at	2017-08-03T14:21:38.701Z	Last contact from a customer, timestamp (DateTime, UTC)
note	null	Internal note for ticket
number	61019	The uniq ticket number
organization_id	null	Id of the organization of a given customer
owner	User	Current owner (agent)
owner_id	1	User id of owner
pending_time	null	Current pending time (DateTime, UTC)
preferences		Sub Hash for special information
priority	2 normal	Ticket priority
priority_id	2	ID of the currently set priority
state	new	Ticket state (new, open. . .)
state_id	1	Ticket state id for available ticket states (new, open. . .)
time_unit	null	Accounted time units for this ticket
title	Feedback Form	Ticket title
type	null	Ticket Type (deprecated)
update_diff_in_min	null	Business hours in minutes within or above the specified SLA for updati
update_escalation_at	null	Time stamp of the last update reaction to the customer (DateTime, UTC
update_in_min	null	Business hours in minutes it took to send the last update response to cus
updated_at	2017-08-03T14:21:38.701Z	Last update timestamp (DateTime, UTC)
updated_by	User	User who updated the ticket
updated_by_id	13	User id of user who updated the ticket

17.2 Article

Field	Sample Value	Description
attachment._name	file1.txt	File name
attachment._content	Hello world	File Content
body	:)	Content of the article
cc	null	Content of the optional cc field
content_type	text/plain	Content type
created_at	2017-08-03T14:21:38.000Z	Article create date (DateTime, UTC)
created_by	See User	Who has created the article
created_by_id	13	Who (UserID) has created the article
from	Christopher Miller via <order@chrispresso.com>	Sender address of the article
id	19	internal (DB) article id
in_reply_to	null	Content of reply to field
internal	FALSE	Is article visible for customer
message_id	null	Message ID (if article was an email)
message_id_md5	null	internal message id MD5 Checksum
origin_by_id	null	For which real user (UserID) the article creation has been done. For example the customer which was calling on the phone
preferences	{ }	Hash for additional information.
references	null	Email references header.
reply_to	null	Content of the reply to field
sender	Customer	Who is the sender (Customer, Agent)
sender_id	2	Which type of user has created the article (Agent, Customer)
subject	Feedback Form	Article subject
ticket_id	19	referencing ticket ID
to	null	Content of the to field
type	web	Article type (phone, email, web...)
type_id	11	Article type id (phone, email, web...)
updated_at	2017-08-03T14:21:38.701Z	Update time of the article (DateTime, UTC)
updated_by	See User	Who has updated the article
updated_by_id	13	Who (UserID) has updated the article

17.3 User

Please note that these fields may vary if you created custom fields (objects) in the admin interface.

Field	Sample Value	Description
active	TRUE	is activ (boolean)
address		User Adress
city		User City
country		User Country
created_at	2017-07-26T21:21:28.000Z	User creation date (DateTime, UTC)
created_by_id	1	ID of user who created the current user
department		User Department
email	chris@chrispresso.com	User E-Mail
fax		User Fax
firstname	Christopher	User Firstname
id	3	Internal id (database, autincrement)
last_login	2017-07-26T21:23:15.019Z	User last login (DateTime, UTC)
lastname	Miller	User Lastname
login	chris@chrispresso.com	User Login
mobile		User Mobile
note		internal note
organization	Chrispresso Inc	Orgnaization name of the current user
organization_id	2	ID which links to the organization name
phone		User Phone
street		User Street
updated_at	2017-07-27T15:04:47.270Z	Last update date (DateTime, UTC)
updated_by_id	3	ID of user who updated the current user
verified	FALSE	is verified (boolean)
vip	FALSE	Is VIP (boolean)
web		User Web Url
zip		User ZIP

CHAPTER 18

Install with Docker-Compose

Warning: We currently do not support Docker environments in productive use.

Docker is a container-based software framework for automating deployment of applications. Compose is a tool for defining and running multi-container Docker applications. This repo is meant to be the starting point for somebody who likes to use dockerized multi-container Zammad in production. The Zammad Docker image uses the stable branch of Zammad's Git repo.

The Docker images are hosted on Dockerhub:

- https://hub.docker.com/r/zammad/zammad-docker-compose/

You need at least 4 GB of RAM to run the containers.

18.1 Install Docker Environment

Your Docker environment needs to be up and running and you need to have docker-compose installed.

18.1.1 Docker

- https://docs.docker.com/engine/installation/

18.1.2 Docker-Compose

- https://docs.docker.com/compose/install/

18.2 Getting started with zammad-docker-compose

18.2.1 Clone GitHub repo

- git clone https://github.com/zammad/zammad-docker-compose.git
- cd zammad-docker-compose

18.2.2 Setting vm.max_map_count for Elasticsearch

- sysctl -w vm.max_map_count=262144

18.2.3 Start Zammad using DockerHub images

- docker-compose up

18.3 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

18.4 Maintenance

18.4.1 Updating Zammad

- docker-compose stop
- git pull
- docker-compose pull
- docker-compose up

18.4.2 Start Zammad building Docker images locally with development branch

- GIT_BRANCH=develop docker-compose -f docker-compose-build.yml up

Recreate locally built images

- GIT_BRANCH=develop docker-compose -f docker-compose-build.yml build –no-cache

18.4.3 Open shell in running Zammad image

- docker-compose exec zammad /bin/bash

18.4.4 Port compatibility error

- The nginx container may have compatibility problems with other machines or services pointing to port 0.0.0.0:80. So to fix this, we'll just have to modify the file *docker-compose.override.yml* and select different ports

CHAPTER 19

Install on Kubernetes via Helm

Warning: We currently do not support Kubernetes installations in productive use.

Kubernetes (k8s) is an open-source system for automating deployment, scaling, and management of containerized applications.

Helm is the package manager for Kubernetes.

This repo is meant to be the starting point for somebody who likes to use dockerized multi-container Zammad on Kubernetes. The Zammad Docker image uses the stable branch of Zammad's Git repo.

The used Docker images are hosted on Dockerhub:

- https://hub.docker.com/r/zammad/zammad-docker-compose/

You need the Helm binary installed / initialized and at least 4 GB of free RAM in the Kubernetes cluster run the containers.

19.1 Add Helm repo

```
helm repo add zammad https://zammad.github.io
```

19.2 Install / Upgrade Zammad

```
helm upgrade --install zammad zammad/zammad --namespace=zammad
```

CHAPTER 20

Updating Zammad

20.1 Source update

Note: Please backup your Zammad instance before update!

20.1.1 1. Download Zammad to your system

You can directly download Zammad from https://ftp.zammad.com/ or use the direct URL to get the latest stable release via https://ftp.zammad.com/zammad-latest.tar.gz

```
root@shell> cd /opt
root@shell> wget https://ftp.zammad.com/zammad-latest.tar.gz
root@shell> tar -C zammad -xzf zammad-latest.tar.gz
root@shell> chown -R zammad /opt/zammad
root@shell> su - zammad
```

20.1.2 2. Install all dependencies

```
zammad@shell> cd zammad
zammad@shell> gem install bundler
```

- For PostgreSQL (note, the option says "without . . . mysql"):

```
zammad@shell> bundle install --without test development mysql
```

- For MySQL (note, the option says "without . . . postgres"):

```
zammad@shell> bundle install --without test development postgres
```

20.1.3 3. Stop Zammad services

Stop the application server, websocket server and scheduler.

20.1.4 4. Upgrade your database

```
zammad@shell> export RAILS_ENV=production
zammad@shell> export RAILS_SERVE_STATIC_FILES=true # only if you use no HTTP reverse␣
↪proxy
zammad@shell> rake db:migrate
zammad@shell> rake assets:precompile
```

20.1.5 5. Start Zammad services

Start the application server, websocket server and scheduler.

20.1.6 6. Go and login to Zammad

20.2 Update with RPM

Note: Please backup your Zammad instance before update!

20.2.1 1. Stop Zammad

```
shell> sudo systemctl stop zammad
```

20.2.2 3. Update Zammad

```
shell> sudo yum update zammad
```

Note: The package will automatically execute maintenance tasks like database changes and will restart Zammad for you.

20.2.3 4. Start Zammad

```
shell> sudo systemctl start zammad
```

20.2.4 5. Go and log in to Zammad

20.3 Update with DEB

Note: Please backup your Zammad instance before update!

20.3.1 1. Stop Zammad

```
shell> sudo systemctl stop zammad
```

20.3.2 3. Update Zammad

```
shell> apt-get update
shell> apt-get upgrade
```

Note: The package will automatically execute maintenance tasks like database changes and will restart Zammad for you.

20.3.3 4. Start Zammad

```
shell> sudo systemctl start zammad
```

20.3.4 5. Go and log in to Zammad

CHAPTER 21

First steps

After installing Zammad, open http://localhost:3000 with your browser and follow the installation wizard.

You can find further information about Zammad's configuration on our Admin-Documentation (for Zammad-Administration within the WebApp).

Basically you need to do the following things:

- Create an admin account
- Set your system basics like hostname and logo
- Connect channels like an email account
- Invite agents and customers to work with you in Zammad

CHAPTER 22

from OTRS

22.1 Install plugins on OTRS

Note: Currently only passwords of OTRS >= 3.3 can be reused in Zammad! Passwords that were stored in another format than the default SHA2 are not possible to use. Users then have to use the password reset procedure.

22.1.1 Install Znuny4OTRS-Repo

This is a dependency of the OTRS migration plugin

- On OTRS 6:
  - https://addons.znuny.com/api/addon_repos/public/1029/latest
- On OTRS 5:
  - https://addons.znuny.com/api/addon_repos/public/615/latest
- On OTRS 4:
  - https://addons.znuny.com/api/addon_repos/public/309/latest
- On OTRS 3:
  - https://addons.znuny.com/api/addon_repos/public/142/latest

22.1.2 Install OTRS migration plugin

- OTRS 6:
  - https://addons.znuny.com/api/addon_repos/public/1085/latest
- OTRS 5:

  - https://addons.znuny.com/api/addon_repos/public/617/latest
- OTRS 4:
  - https://addons.znuny.com/api/addon_repos/public/383/latest
- OTRS 3.1 - 3.3:
  - https://addons.znuny.com/api/addon_repos/public/287/latest

22.2 Import via Browser

After installing Zammad, open http://localhost:3000 with your browser and follow the installation wizard. From there you're able to start the migration from OTRS.

See the Video at this site .

22.3 Import via command line

If you miss this at the beginning or you want to re-import again you have to use the command line at the moment.

Stop all Zammad processes and switch Zammad to import mode (no events are fired - e. g. notifications, sending emails, . . .)

22.3.1 If you installed the Zammad DEB or RPM package

```
zammad run rails c
```

22.3.2 If you installed from source

```
su zammad
cd /opt/zammad
rails c
```

22.3.3 Enter the following commands in the rails console

```
Setting.set('import_otrs_endpoint', 'http://xxx/otrs/public.pl?Action=ZammadMigrator')
Setting.set('import_otrs_endpoint_key', 'xxx')
Setting.set('import_mode', true)
Import::OTRS.start
```

After the import is done switch Zammad back to non-import mode and mark the system initialization as done.

```
Setting.set('import_mode', false)
Setting.set('system_init_done', true)
```

Start all Zammad processes again. Done.

22.4 Importing a diff

In some cases it might be desirable to update the already imported data from OTRS. This is possible with the following commands.

22.4.1 Enter the following commands in the rails console

```
Setting.set('import_otrs_endpoint', 'http://xxx/otrs/public.pl?Action=ZammadMigrator')
Setting.set('import_otrs_endpoint_key', 'xxx')
Setting.set('import_mode', true)
Setting.set('system_init_done', false)
Import::OTRS.diff_worker
```

After the import is done switch Zammad back to non-import mode and mark the system initialization as done.

```
Setting.set('import_mode', false)
Setting.set('system_init_done', true)
```

Start all Zammad processes again. Done.

22.5 Restarting from scratch

First make sure all Zammad processes are stopped. After that reset your database.

22.5.1 If you installed the Zammad DEB or RPM package

```
zammad run rake db:drop
zammad run rake db:create
zammad run rake db:migrate
zammad run rake db:seed
```

22.5.2 If you installed from source

```
rake db:drop
rake db:create
rake db:migrate
rake db:seed
```

After that your DB is reset and you can start the import right over.

CHAPTER 23

from Zendesk

Sorry, this still needs to be added :-(

Do you want to contribute to the Zammad documentation?

Open a new GitHub pull request at https://github.com/zammad/zammad-documentation with your changes.

CHAPTER 24

Console

Zammad uses Ruby on Rails so you can make use of the rails console: http://guides.rubyonrails.org/command_line.html

To open the rails console on the shell you have to enter the following commands.

24.1 Start Zammad's Rails console

24.1.1 If you used a Zammad DEB or RPM package:

```
shell> zammad run rails c
```

24.1.2 If you installed Zammad from source use:

```
shell> rails c
```

24.2 Example commands

24.2.1 Find user

```
rails> User.find(4)
rails> User.find_by(email: 'your@email')
```

24.2.2 Find group

```
rails> Group.find_by(name: 'Users').follow_up_possible
```

24.2.3 Change / Update E-Mail-Adress of User

```
rails> u=User.find(**USERID**)
rails> u.email = 'user@exmaple.com'
rails> u.save!
```

You need to find the User-ID of the user first for this.

24.2.4 Change / Update Login name of User

```
rails> u=User.find(**USERID**)
rails> u.login = 'user@exmaple.com'
rails> u.save!
```

You need to find the User-ID of the user first for this.

24.2.5 Update the customer of a bundle of tickets

```
rails> Ticket.where(customer_id: 4).update_all(customer_id: 1)
```

24.2.6 Change priority

```
rails> priority2 = Ticket::Priority.find(2)
rails> priority2.name = '2-high'
rails> priority2.default_create = true
rails> priority2.save!
```

24.2.7 Get ticket_hook setting

```
rails> Setting.get('ticket_hook')
```

24.2.8 Get fqdn setting

```
rails> Setting.get('fqdn')
```

24.2.9 Find storage_provide setting

```
rails> Setting.find_by(name: 'storage_provider')
```

24.2.10 Set storage_rpovider Setting

```
rails> Setting.set('storage_provider', 'DB')
```

24.2.11 Fetch mails

```
rails> Channel.fetch
```

24.2.12 Get ticket state types

```
rails> Ticket::StateType.all
```

24.2.13 Add new ticket state

option a) a pending remidner state (send reminder notification to agent if time has reached)

```
rails>
   Ticket::State.create_or_update(
     name: 'pending customer feedback',
     state_type: Ticket::StateType.find_by(name: 'pending reminder'),
     ignore_escalation: true,
     created_by_id: 1,
     updated_by_id: 1,
   )
```

option b) a pending action state (convert ticket into next state if time has reached)

```
rails>
   Ticket::State.create_or_update(
     name: 'pending and reopen',
     state_type: Ticket::StateType.find_by(name: 'pending action'),
     ignore_escalation: true,
     next_state: Ticket::StateType.find_by(name: 'open'),
     created_by_id: 1,
     updated_by_id: 1,
   )
```

to make them available in UI you need to execute the following:

```
rails>
   attribute = ObjectManager::Attribute.get(
     object: 'Ticket',
     name: 'state_id',
   )
   attribute.data_option[:filter] = Ticket::State.by_category(:viewable).pluck(:id)
   attribute.screens[:create_middle]['ticket.agent'][:filter] = Ticket::State.by_
↪category(:viewable_agent_new).pluck(:id)
   attribute.screens[:create_middle]['ticket.customer'][:filter] = Ticket::State.by_
↪category(:viewable_customer_new).pluck(:id)
   attribute.screens[:edit]['ticket.agent'][:filter] = Ticket::State.by_
↪category(:viewable_agent_new).pluck(:id)
```

(continues on next page)

(continued from previous page)

```
  attribute.screens[:edit]['ticket.customer'][:filter] = Ticket::State.by_
↪category(:viewable_customer_edit).pluck(:id)
  attribute.save!
```

24.2.14 Delete a certain ticket

```
rails> Ticket.find(4).destroy
```

24.2.15 Delete some tickets

```
rails> tickets_to_keep = [1, 2, 3] # enter the ids of all tickets you want to keep
rails> (Ticket.all.pluck(:id) - tickets_to_keep).each { |id| Ticket.find(id).destroy }
```

24.2.16 Delete all tickets

```
rails> Ticket.destroy_all
```

24.2.17 Add translation

```
rails> Translation.create_if_not_exists( :locale => 'de-de', :source => "New",␣
↪:target => "Neu", format: 'string', created_by_id: 1, updated_by_id: 1 )
```

24.2.18 Set admin rights for user

```
rails> u = User.find_by(email: 'you@example.com')
rails> u.roles = Role.where(name: ['Agent', 'Admin'])
rails> u.save!
```

24.2.19 Set password for user

```
rails> User.find_by(email: 'you@example.com').update!(password: 'your_new_password')
```

24.2.20 Configuring Elasticsearch

```
rails> Setting.set('es_url', 'http://127.0.0.1:9200')
rails> Setting.set('es_user', 'elasticsearch')
rails> Setting.set('es_password', 'zammad')
rails> Setting.set('es_index', Socket.gethostname + '_zammad')
rails> Setting.set('es_attachment_ignore', [ '.png', '.jpg', '.jpeg', '.mpeg', '.mpg',
↪ '.mov', '.bin', '.exe', '.box', '.mbox' ] )
rails> Setting.set('es_attachment_max_size_in_mb', 50)
```

24.2.21 Use the OTRS importer from the shell

```
rails> Setting.set('import_otrs_endpoint', 'http://xxx/otrs/public.pl?
↪Action=ZammadMigrator')
rails> Setting.set('import_otrs_endpoint_key', 'xxx')
rails> Setting.set('import_mode', true)
rails> Import::OTRS.start
```

24.2.22 Enable proxy

```
rails> Setting.set('proxy', 'proxy.example.com:3128')
rails> Setting.set('proxy_username', 'some user')
rails> Setting.set('proxy_password', 'some pass')
```

24.2.23 Destroy stuff

```
rails> OnlineNotification.destroy_all
rails> ActivityStream.destroy_all
rails> RecentView.destroy_all
rails> History.destroy_all
```

24.2.24 Fill a testsystem with testdata (don´t do this on your production system!)

```
rails> FillDB.load(agents: 50,customers: 1000,groups: 20,organizations: 40,overviews:␣
↪5,tickets: 100,)
```

CHAPTER 25

Start

We would be glad if you contribute to Zammad. You can do this in several ways. Contributions are mainly done by forking one of our repos on GitHub and creating a pull request with your changes.

All repos can be found at https://github.com/zammad

25.1 Source Code

The Zammad source code can be found on GitHub at https://github.com/zammad/zammad

25.2 Documentation

Do you want to contribute to the Zammad documentation?

Open a new GitHub pull request at https://github.com/zammad/zammad-documentation with your changes.

The Zammad documentation is hosted on readthedocs.org. You can read it there at https://docs.zammad.org or browse the files via GitHub which also renders the used ReStructuredText markup.

25.2.1 ReStructuredText markup

If you like to edit the docs use the ReStructuredText markup language. Information about this language can be found at:

- http://www.sphinx-doc.org/en/stable/rest.html
- http://docutils.sourceforge.net/docs/user/rst/quickref.html
- http://docs.readthedocs.io/en/latest/_themes/sphinx_rtd_theme/demo_docs/source/demo.html

Thanks!

Zammad Team

CHAPTER 26

Branches

The Zammad main repo at https://github.com/zammad/zammad has several branches

26.1 Master

- Current unreleased development state of next stable minor release
- Bug fixes of current stable version are added here
- Is the branch where features work correctly
- Could be used for production environment by experienced users
- If current stable version is 1.1.0 this will become 1.1.1

26.2 Develop

- Default GitHub branch
- Current unreleased development state of next major release
- Is the first instance where all features are being developed
- This branch will have open issues
- If current stable version is 1.1.0 this will become 1.2.0
- Unstable!
- Should not be used in production environment!

26.3 Stable

- Current stable release
- Can be used for production
- Stable bugfixes will be merged from master when new stable minor version will be released

26.4 Stable-X.x

- There will be several more stable branches because we'll support the last three major versions of Zammad
- If current stable version is 1.2.0 then the name of the branch is stable-1.2 and there also would be stable-1.1 and stable-1.0

CHAPTER 27

Packages

- Zammad packages are built on packager.io.
- You can find all Zammad packages at https://packager.io/gh/zammad/zammad
- Builds of new packages are triggered with every push to our GitHub repo
- If you fork the Zammad repo you can use packager.io to get builds for your fork
- Just change the file ".pkgr.yml" to fit your needs

CHAPTER 28

Continuous integration

All pushes to our main repo at https://github.com/zammad/zammad will trigger build tests. We use internal build tests on internal and external continuous integration platforms.

28.1 Internal

- Done on our private continuous integration platform

28.2 External

28.2.1 Travis-CI

- You can find the build test results at https://travis-ci.org/zammad/zammad
- If you fork the Zammad repo you're able to also use travis-ci.org to get your builds tested
- Just change the file ".travis.yml" to fit your needs
- Current build test status is:

CHAPTER 29

Code quality

- All pushes to our main repo at https://github.com/zammad/zammad will be checked by several libraries and services

29.1 Rubocop

- Code is being checked by Rubocop
- http://rubocop.readthedocs.io

29.2 Codeclimate

- Code is also being checked on https://codeclimate.com.
- You can find the results at https://codeclimate.com/github/zammad/zammad
- If you fork the Zammad repo you can use codeclimate.com to check your code
- Just change the file ".codeclimate.yml" to fit your needs

CHAPTER 30

Install with Docker

Docker is a container-based software framework for automating deployment of applications. Our Docker image is a **single container** based application designed to have Zammad **up and running fast for testing purposes**.

Please note that this is a non persistent storage container and **all Zammad data is lost** when you're stopping the container.

If you like to run Docker in production environment try our Docker-compose version: *Install with Docker-Compose* .

Your Docker environment needs to be up and running.

You can find the image at https://hub.docker.com/r/zammad/zammad/

You need at least 4 GB of RAM to run the container.

30.1 Run the Docker Container

Docker run will run a command in a new container, -i attaches stdin and stdout, -t allocates a tty.

30.1.1 Set vm.max_map_count for Elasticsearch

```
sysctl -w vm.max_map_count=262144
```

30.1.2 Run docker container

```
docker container run -ti --rm --name zammad -p 80:80 zammad/zammad
```

That's it! You're now using a bash shell inside of a Zammad docker container using the develop branch of the GitHub repo.

To disconnect or detach from the shell without exiting, use the escape sequence Ctrl-p + Ctrl-q.

30.2 Go to http://localhost and you'll see:

- "Welcome to Zammad!", there you need to create your admin user.

CHAPTER 31

Install with Vagrant

Vagrant is a tool for building complete development environments. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time, increases development/production parity, and makes the "works on my machine" excuse a relic of the past.

Be aware that Vagrant is meant for developers and therefore uses our unstable packages from the "develop" branch on GitHub.

Let's begin using Vagrant! First be sure that a Vagrant provider is installed. You can use "Virtual Box" from https://www.virtualbox.org .

31.1 Clone the Vagrant file

```
git clone git@github.com:zammad/zammad-vagrant.git
cd zammad-vagrant
```

31.2 Run Vagrant

31.2.1 For stable branch package

```
PACKAGER_REPO=stable vagrant up --provision
```

31.2.2 For develop branch package

```
vagrant up --provision
```

That's it! You're now running Zammad in a Vagrant environment.

31.3 Go to http://localhost:8080 and you'll see:

- "Welcome to Zammad!", there you need to create your admin user and invite other agents.

31.4 SSH into the machine

After "vagrant up"

```
vagrant ssh
```

After this you can switch to root user via:

```
sudo -i
```

31.5 Problems starting the VM?

If you get errors like:

```
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'centos/7' is up to date...
==> default: VirtualBox VM is already running.
```

Use the following commands to fix it:

```
vboxmanage controlvm Zammad poweroff
```

CHAPTER 32

Introduction

Zammad is a web based open source helpdesk/ticket system with many features to manage customer communication via several channels like telephone, facebook, twitter, chat and e-mails.

This chapter describes the Zammad API v1.

32.1 The API

Zammad provides a REST/JSON API. Its endpoints are documented with the HTTP method for the request and a partial resource.

Example:

```
GET /api/v1/users
```

The full URL looks like:

```
https://your_zammad/api/v1/users
```

Curly braces { } indicate values you have to supply for the URL.

Example:

```
GET  /api/v1/users/{id}
```

32.2 Authentication

Zammad supports three different authentication methods for API.

32.2.1 HTTP Basic Authentication (username/password)

The username/password must be provided as HTTP header in the HTTP call. The Zammad admin can enable/disable the authentication method in the admin interface. Read more about HTTP basic authentication [here](https://en.wikipedia.org/wiki/Basic_access_authentication).

Example:

```
curl -u {username}:{password} https://your_zammad/api/v1/users
```

32.2.2 HTTP Token Authentication (access token)

The access token must be provided as HTTP header in the HTTP call. Each user needs to create its own access token in the user preferences. The Zammad admin can enable/disable the authentication method in the admin interface.

Example:

```
curl -H "Authorization: Token token={your_token}" https://your_zammad/api/v1/users
```

32.2.3 OAuth2 (token access)

The Zammad API supports OAuth2 authorization. In order to create OAuth2 tokens for an external application, the Zammad user needs to create an application in the admin interface. The access token then has to be given within the HTTP header:

Example:

```
curl -H "Authorization: Bearer {your_token}" https://your_zammad/api/v1/users
```

32.3 Request Format

Zammad uses JSON for its API, so you need to set a "Content-Type: application/json" in each HTTP call. Otherwise the response will be text/html.

Example:

```
POST /api/v1/users/{id} HTTP/1.1
Content-Type: application/json

{
 "name":"some name",
 "organization_id": 123,
 "note":"some note"
}
```

32.4 Example CURL Requests

Get information:

```
curl -u test@zammad.com:test123 https://xxx.zammad.com/api/v1/tickets/3
```

Put information:

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -X PUT -d '{␣
↪json: "data" }' https://xxx.zammad.com/api/v1/tickets/3
```

Post information:

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -X POST -d '{␣
↪json: "data" }' https://xxx.zammad.com/api/v1/tickets/3
```

32.5 Example CURL Requests (for tickets and users)

Create a new ticket:

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -X POST -d '{
↪"title":"Help me!","group": "Users","article":{"subject":"some subject","body":
↪"some message","type":"note","internal":false},"customer":"email_of_existing_
↪customer@example.com","note": "some note"}' https://xxx.zammad.com/api/v1/tickets
```

Search for tickets (with contains "some message"):

```
curl -u test@zammad.com:test123 'https://xxx.zammad.com/api/v1/tickets/search?
↪query=some+message&limit=10&expand=true'
```

Search for tickets (for tickets with state new and open):

```
curl -u test@zammad.com:test123 'https://xxx.zammad.com/api/v1/tickets/search?
↪query=state:new%20OR%20state:open&limit=10&expand=true'
```

For more search examples regarding searching, please see this page .

Create an new user:

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -X POST -d '{
↪"firstname":"Bob","lastname":"Smith","email":"email_of_customer@example.com","roles
↪":["Customer"],"password":"some_password"}' https://xxx.zammad.com/api/v1/users
```

Create an new user (with welcome email):

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -X POST -d '{
↪"firstname":"Bob","lastname":"Smith","email":"email_of_customer@example.com","roles
↪":["Customer"],"password":"some_password","invite":true}' https://xxx.zammad.com/
↪api/v1/users
```

Search for users:

```
curl -u test@zammad.com:test123 'https://xxx.zammad.com/api/v1/users/search?
↪query=smith&limit=10&expand=true'
```

32.6 Example CURL Request on behalf of a different user

It is possible to do a request on behalf of a different user. If you have your own application and you want to create a ticket for the customer without the information that the api user has created this ticket then you can transfer the target user with the request to create the ticket on behalf of the customer user:

```
curl -u test@zammad.com:test123 -H "Content-Type: application/json" -H "X-On-Behalf-
↪Of: user-login" -X POST -d '{"title":"Help me!","group": "Users","article":{"subject
↪":"some subject","body":"some message","type":"note","internal":false},"customer":
↪"email_of_existing_customer@example.com","note": "some note"}' https://xxx.zammad.
↪com/api/v1/tickets
```

The value of the header has to contain one of the following values:

- user id
- user login
- user email

The value types will be checked in a cascade and the first detected user by id, login or email will be used for the request action.

This functionality can be used for any type of action.

Requirements for the feature:

- Authenticated user must have **admin.user** permissions
- Feature is available since Zammad version 2.4

32.7 Response Format

If a response is successful, an HTTP status code in the 200 or 300 range will be returned. If an item has been created or updated, all new attributes will be returned (also server side generated attributes like created_at and updated_at).

Example:

```
Status: 201 Created
Content-Type:application/json; charset=utf-8

{
 "id": 123,
 "name":"some name",
 "organization_id": 123,
 "note":"some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

32.8 Response Format (expanded)

If you want to retrieve expanded information for a request (e. g. the organization attribute), you just need to add an `expand=true` to the request URL.

Example:

```
GET /api/v1/users/{id}?expand=true HTTP/1.1
```

will return the following structure, expanded by “organization”:

```
Status: 200 Ok
Content-Type:application/json; charset=utf-8

{
 "id": 123,
 "name":"some name",
 "organization_id": 123,
 "organization": "Some Organization Name",
 "note":"some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

32.9 Pagination

All resources support pagination:

```
GET /api/v1/users?expand=true&page=1&per_page=5 HTTP/1.1
```

will return five records beginning with first record of all:

```
Status: 200 Ok
Content-Type:application/json; charset=utf-8

[
 {
   "id": 1,
   "name":"some name 1",
   "organization_id": 123,
   "organization": "Some Organization Name",
   "note":"some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 2,
   "name":"some name 2",
   "organization_id": 345,
   "organization": "Some Other Organization Name",
   "note":"some note",
   "updated_at": "2016-08-17T07:55:42.221Z",
   "created_at": "2016-08-16T09:112:42.221Z"
 },
 ...
}
```

32.10 API clients

- Ruby Client - https://github.com/zammad/zammad-api-client-ruby
- PHP Client - https://github.com/zammad/zammad-api-client-php
- .NET Client - https://github.com/Asesjix/Zammad-Client

CHAPTER 33

User

33.1 me - current user

Required permission:

- any (only valid authentication)

Request:

```
GET /api/v1/users/me
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "firstname": "Bob",
 "lastname": "Smith",
 "email": "bob@smith.example.com",
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
},
```

33.2 List

Required permission:

- ticket.agent or admin.user (can read all users)
- any (can only read its own user if exists)

Request:

```
GET /api/v1/users
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "firstname": "Bob",
   "lastname": "Smith",
   "email": "bob@smith.example.com",
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "firstname": "Martha",
   "lastname": "Braun",
   "email": "marta@braun.example.com",
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

33.3 Search

Required permission:

- ticket.agent or admin.user (can read all users)

Request:

```
GET /api/v1/users/search?query=what&limit=10
```

Note: As of Zammad 2.6 parameters (sort_by=some_row and order_by=asc or desc) can also be used for sorting.

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "firstname": "Bob",
   "lastname": "Smith",
   "email": "bob@smith.example.com",
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
```

(continues on next page)

(continued from previous page)

```
    "created_at": "2016-08-16T07:55:42.119Z"
  },
  {
    "id": 124,
    "firstname": "Martha",
    "lastname": "Braun",
    "email": "marta@braun.example.com",
    ...
    "note": "some note",
    "updated_at": "2016-08-16T07:55:42.119Z",
    "created_at": "2016-08-16T07:55:42.119Z"
  },
]
```

33.4 Show

Required permission:

- ticket.agent or admin.user (can read all users)
- customer with same organization (can read all users of same organization)
- any (can only read it's own user if exists)

Request:

```
GET /api/v1/users/{id}
```

Response:

```
Status: 200 Ok

{
  "id": 123,
  "firstname": "Bob",
  "lastname": "Smith",
  "email": "bob@smith.example.com",
  ...
  "note": "some note",
  "updated_at": "2016-08-16T07:55:42.119Z",
  "created_at": "2016-08-16T07:55:42.119Z"
}
```

33.5 Create

Required permission:

- admin.user
- ticket.agent (can not set roles/role_ids and not set groups/group_ids - roles.default_at_signup roles will get assigned automatically)
- any - until user_create_account is disabled (can not set roles/role_ids and not set groups/group_ids - roles.default_at_signup roles will get assigned automatically)

Request:

```
POST /api/v1/users

{
 "firstname": "Bob",
 "lastname": "Smith",
 "email": "bob@smith.example.com",
 "organization": "Some Organization Name",
 ...
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "firstname": "Bob",
 "lastname": "Smith",
 "email": "bob@smith.example.com",
 "organization_id": 123,
 "organization": "Some Organization Name",
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

33.6 Update

Required permission:

- admin.user
- ticket.agent (can only update customer accounts and not set roles/role_ids and not set groups/group_ids - already assigned attributes will not changed)

Request:

```
PUT /api/v1/users/{id}

{
 "firstname": "Bob",
 "lastname": "Smith",
 "email": "bob@smith.example.com",
 "organization": "Some Other Organization Name",
 ...
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
```

(continues on next page)

(continued from previous page)

```
 "firstname": "Bob",
 "lastname": "Smith",
 "email": "bob@smith.example.com",
 "organization_id": 124,
 "organization": "Some Other Organization Name",
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

33.7 Delete

Required permission:

- admin.user (only if no references in history tables and tickets exist)

Request:

```
DELETE /api/v1/users/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 34

Organization

34.1 List

Required permission:

- ticket.agent or admin.organization (can read all organizations)
- any (can only read its own organization if exists)

Request:

```
GET /api/v1/organizations
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "name": "Org 1",
   "shared": true,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "name": "Org 2",
   "shared": false,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
```

(continues on next page)

(continued from previous page)

```
 },
]
```

34.2 Search

Required permission:

- ticket.agent or admin.organization (can read all organization)

Request:

```
GET /api/v1/organizations/search?query=what&limit=10
```

Note: As of Zammad 2.6 parameters (sort_by=some_row and order_by=asc or desc) can also be used for sorting.

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "name": "Org 1",
   "shared": true,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "name": "Org 2",
   "shared": false,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

34.3 Show

Required permission:

- ticket.agent or admin.organization (can read all organizations)
- any (can only read its own user if exists)

Request:

```
GET /api/v1/organizations/{id}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Org 1",
 "shared": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

34.4 Create

Required permission:

- admin.organization

Request:

```
POST /api/v1/organizations

{
 "name": "Org 1",
 "shared": true,
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "name": "Org 1",
 "shared": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

34.5 Update

Required permission:

- admin.organization

Request:

```
PUT /api/v1/organizations/{id}

{
```

(continues on next page)

(continued from previous page)

```
 "id": 123,
 "name": "Org 1",
 "shared": true,
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Org 1",
 "shared": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

34.6 Delete

Required permission:

- admin.organization (only if no references in history tables and tickets exist)

Request:

```
DELETE /api/v1/organization/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 35

Group

35.1 List

Required permission:

- admin.group (can read all groups)

Request:

```
GET /api/v1/groups
```

Response:

```
Status: 200 Ok

[
  {
    "id": 123,
    "name": "Group 1",
    "signature_id": 123,
    "email_address_id": 123,
    "assignment_timeout": 180,
    "follow_up_possible": "yes",
    "follow_up_assignment": true,
    "active": true,
    "note": "some note",
    "updated_at": "2016-08-16T07:55:42.119Z",
    "created_at": "2016-08-16T07:55:42.119Z"
  },
  {
    "id": 124,
    "name": "Group 2",
    "signature_id": 123,
    "email_address_id": 123,
    "assignment_timeout": 180,
```

(continues on next page)

(continued from previous page)

```
    "follow_up_possible": "no",
    "follow_up_assignment": false,
    "active": true,
    "note": "some note",
    "updated_at": "2016-08-16T07:55:42.119Z",
    "created_at": "2016-08-16T07:55:42.119Z"
  },
]
```

35.2 Show

Required permission:

- admin.group (can read all groups)

Request:

```
GET /api/v1/groups/{id}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Group 1",
 "signature_id": 123,
 "email_address_id": 123,
 "assignment_timeout": 180,
 "follow_up_possible": "yes",
 "follow_up_assignment": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

35.3 Create

Required permission:

- admin.group

Request:

```
POST /api/v1/groups


{
 "name": "Group 1",
 "signature_id": 123,
 "email_address_id": 123,
 "assignment_timeout": 180,
```

(continues on next page)

(continued from previous page)

```
 "follow_up_possible": "yes",
 "follow_up_assignment": true,
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "name": "Group 1",
 "signature_id": 123,
 "email_address_id": 123,
 "assignment_timeout": 180,
 "follow_up_possible": "yes",
 "follow_up_assignment": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

35.4 Update

Required permission:

- admin.group

Request:

```
PUT /api/v1/groups/{id}

{
 "id": 123,
 "name": "Group 1",
 "signature_id": 123,
 "email_address_id": 123,
 "assignment_timeout": 180,
 "follow_up_possible": "yes",
 "follow_up_assignment": true,
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Group 1",
 "signature_id": 123,
 "email_address_id": 123,
```

(continues on next page)

(continued from previous page)

```
 "assignment_timeout": 180,
 "follow_up_possible": "yes",
 "follow_up_assignment": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

35.5 Delete

Required permission:

- admin.group (only if no references in history tables and tickets exist)

Request:

```
DELETE /api/v1/groups/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 36

Ticket

36.1 List

Required permission:

- ticket.agent (access to all ticket in allocated groups)
- ticket.customer (access to all ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
GET /api/v1/tickets
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "title": "Help me!",
   "group_id": 1,
   "state_id": 1,
   "priority_id": 2,
   "customer_id": 2,
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "title": "Just want to ask for support",
   "state_id": 2,
```

(continues on next page)

(continued from previous page)

```
   "priority_id": 2,
   "customer_id": 2,
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

36.2 Search

Required permission:

- ticket.agent (access to all ticket in allocated groups)
- ticket.customer (access to all ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
GET /api/v1/tickets/search?query=what&limit=10
```

Note: As of Zammad 2.6 parameters (sort_by=some_row and order_by=asc or desc) can also be used for sorting.

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "title": "Help me!",
   "group_id": 1,
   "state_id": 1,
   "priority_id": 2,
   "customer_id": 2,
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "title": "Just want to ask for support",
   "state_id": 2,
   "priority_id": 2,
   "customer_id": 2,
   ...
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

36.3 Show

Required permission:

- ticket.agent (access to all ticket in allocated groups)
- ticket.customer (access to all ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
GET /api/v1/tickets/{id}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "title": "Help me!",
 "group_id": 1,
 "state_id": 1,
 "priority_id": 2,
 "customer_id": 2,
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

36.4 Create

Required permission:

- ticket.agent (create in all allocated groups)
- ticket.customer

Request:

```
POST /api/v1/tickets

{
 "title": "Help me!",
 "group": "Users",
 "customer": "email_of_existing_customer@example.com",
 "article": {
   "subject": "some subject",
   "body": "some message",
   "type": "note",
   "internal": false
 },
 ...
 "note": "some note"
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "title": "Help me!",
 "group_id": 1,
 "state_id": 1,
 "priority_id": 2,
 "customer_id": 2,
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

For more article attributes have a look into “Ticket Article”.

If you want to include attachments of the first article, the payload looks like:

Request:

```
POST /api/v1/tickets

{
 "title": "Help me!",
 "group": "Users",
 "article": {
   "subject": "some subject",
   "body": "some message",
   "attachments": [
     {
       "filename": "some_file1.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     },
     {
       "filename": "some_file2.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     }
   ]
 },
 ...
 "note": "some note"
}
```

If you want to add inline images, just use data URIs in HTML markup:

Request:

```
POST /api/v1/tickets

{
 "title": "Help me!",
 "group": "Users",
 "article": {
   "content_type": "text/html",
   "subject": "some subject",
```

(continues on next page)

(continued from previous page)

```
   "body": "<b>some</b> message witn inline image <img src=\"
↪ABCDEFG==\">"
 },
 ...
 "note": "some note"
}
```

If you want to use or create an customer by email address at ticket creation, you can do with "guess:customer@example.com" in the customer_id attribute:

Request:

```
POST /api/v1/tickets

{
 "title": "Help me!",
 "group": "Users",
 "customer_id": "guess:customer@example.com",
 ...
 "note": "some note"
}
```

36.5 Update

Required permission:

- ticket.agent (access to all ticket in allocated groups)
- ticket.customer (access to all ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
PUT /api/v1/tickets/{id}

{
 "id": 123,
 "title": "Help me!",
 "group": "Users",
 "state": "open",
 "priority": "3 high",
 "article": {
   "subject": "some subject of update",
   "body": "some message of update"
 },
 ...
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "title": "Help me!",
```

(continues on next page)

(continued from previous page)

```
 "group_id": 1,
 "state_id": 1,
 "priority_id": 2,
 ...
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

If you want to include attachments of the article, the payload looks like:

Request:

```
PUT /api/v1/tickets/{id}

{
 "id": 123,
 "title": "Help me!",
 "group": "Users",
 "article": {
   "subject": "some subject",
   "body": "some message",
   "attachments": [
     {
       "filename": "some_file1.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     },
     {
       "filename": "some_file2.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     }
   ]
 },
 ...
 "note": "some note"
}
```

If you want to add inline images, just use data URIs in HTML markup:

Request:

```
PUT /api/v1/tickets/{id}

{
 "id": 123,
 "title": "Help me!",
 "group": "Users",
 "article": {
   "content_type": "text/html",
   "subject": "some subject",
   "body": "<b>some</b> message witn inline image <img src=\"
↪ABCDEFG==\">"
 },
 ...
 "note": "some note"
}
```

36.6 Delete

Required permission:

- admin

Request:

```
DELETE /api/v1/tickets/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 37

Ticket State

37.1 List

Required permission:

- admin.object (can read all ticket states)
- ticket.agent (can read all ticket states)
- ticket.customer (can read all ticket states)

Request:

```
GET /api/v1/ticket_states
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "name": "Ticket State 1",
   "state_type_id": 1,
   "next_state_id": null,
   "ignore_escalation": true,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "name": "Ticket State 2",
   "state_type_id": 2,
```

(continues on next page)

(continued from previous page)

```
   "next_state_id": 4,
   "ignore_escalation": false,
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

37.2 Show

Required permission:

- admin.object (can read all ticket states)
- ticket.agent (can read all ticket states)
- ticket.customer (can read all ticket states)

Request:

```
GET /api/v1/ticket_states/{id}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Ticket State 1",
 "state_type_id": 1,
 "next_state_id": null,
 "ignore_escalation": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

37.3 Create

Required permission:

- admin.object

Request:

```
POST /api/v1/ticket_states

{
 "name": "Ticket State 1",
 "state_type_id": 1,
 "next_state_id": null,
 "ignore_escalation": true,
```

(continues on next page)

(continued from previous page)

```
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "name": "Ticket State 1",
 "state_type_id": 1,
 "next_state_id": null,
 "ignore_escalation": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

37.4 Update

Required permission:

- admin.object

Request:

```
PUT /api/v1/ticket_states/{id}

{
 "id": 123,
 "name": "Ticket State 1",
 "state_type_id": 1,
 "next_state_id": null,
 "ignore_escalation": true,
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Ticket State 1",
 "state_type_id": 1,
 "next_state_id": null,
 "ignore_escalation": true,
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

37.5 Delete

Required permission:

- admin.object (only if no references in history tables and tickets exist)

Request:

```
DELETE /api/v1/ticket_states/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 38

Ticket Priority

38.1 List

Required permission:

- admin.object (can read all ticket states)
- ticket.agent (can read all ticket states)
- ticket.customer (can read all ticket states)

Request:

```
GET /api/v1/ticket_priorities
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "name": "Ticket Priority 1",
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
 {
   "id": 124,
   "name": "Ticket Priority 2",
   "active": true,
   "note": "some note",
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
```

(continues on next page)

(continued from previous page)

```
 },
]
```

38.2 Show

Required permission:

- admin.object (can read all ticket states)
- ticket.agent (can read all ticket states)
- ticket.customer (can read all ticket states)

Request:

```
GET /api/v1/ticket_priorities/{id}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Ticket Priority 1",
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

38.3 Create

Required permission:

- admin.object

Request:

```
POST /api/v1/ticket_priorities

{
 "name": "Ticket Priority 1",
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 201 Created

{
 "id": 123,
 "name": "Ticket Priority 1",
```

(continues on next page)

(continued from previous page)

```
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

38.4 Update

Required permission:

- admin.object

Request:

```
PUT /api/v1/ticket_priorities/{id}

{
 "id": 123,
 "name": "Ticket Priority 1",
 "active": true,
 "note": "some note"
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "name": "Ticket Priority 1",
 "active": true,
 "note": "some note",
 "updated_at": "2016-08-16T07:55:42.119Z",
 "created_at": "2016-08-16T07:55:42.119Z"
}
```

38.5 Delete

Required permission:

- admin.object (only if no references in history tables and tickets exist)

Request:

```
DELETE /api/v1/ticket_priorities/{id}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 39

Ticket Article

39.1 By Ticket

Required permission:

- ticket.agent (access to related ticket)
- ticket.customer (access to related ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
GET /api/v1/ticket_articles/by_ticket/{ticketId}
```

Response:

```
Status: 200 Ok

[
 {
   "id": 3,
   "ticket_id": 3,
   "from": "Bob Smith",
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
   "content_type": "text/html",
   "type": "note",
   "internal": false,
   "time_unit": "12.0"
   ...
   "updated_at": "2016-08-15T07:55:42.119Z",
   "created_at": "2016-08-15T07:55:42.119Z"
},
```

(continues on next page)

(continued from previous page)

```
{
   "id": 4,
   "ticket_id": 3,
   "from": "Bob Smith",
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
   "content_type": "text/html",
   "type": "note",
   "internal": false,
   "time_unit": "15.0"
   ...
   "updated_at": "2016-08-16T07:55:42.119Z",
   "created_at": "2016-08-16T07:55:42.119Z"
 },
]
```

39.2 Show

Required permission:

- ticket.agent (access to related ticket)
- ticket.customer (access to related ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
GET /api/v1/ticket_articles/{id}
```

Response:

```
Status: 200 Ok

{
   "id": 3,
   "ticket_id": 3,
   "from": "Bob Smith",
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
   "content_type": "text/html",
   "type": "note",
   "internal": false,
   "time_unit": "12.0"
   "attachments": [
     {
       "id": 123,
       "filename": "some_file1.txt",
       "preferences": {
         "Mime-Type": "text/plain"
       }
     },
```

(continues on next page)

(continued from previous page)

```
    {
      "id": 124,
      "filename": "some_file2.txt",
      "preferences": {
        "Mime-Type": "text/plain"
      }
    }
  ],
  ...
  "created_at": "2016-10-19T10:07:12.011Z",
  "updated_at": "2017-01-18T12:45:53.420Z"
}
```

39.3 Create

Required permission:

- ticket.agent (access to related ticket)
- ticket.customer (access to related ticket with customer_id ** current_user.id || organization_id ** current_user.organization_id)

Request:

```
POST /api/v1/ticket_articles

{
  "ticket_id": 3,
  "to": "",
  "cc": "",
  "subject": "some subject",
  "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
  "content_type": "text/html",
  "type": "note",
  "internal": false,
  "time_unit": "12"
}
```

Response:

```
Status: 201 Created

{
  "id": 3,
  "ticket_id": 3,
  "from": "Bob Smith",
  "to": "",
  "cc": "",
  "subject": "some subject",
  "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
  "content_type": "text/html",
  "type": "note",
  "internal": false,
  "time_unit": "12.0"
  ...
```

(continues on next page)

(continued from previous page)

```
   "created_at": "2016-10-19T10:07:12.011Z",
   "updated_at": "2017-01-18T12:45:53.420Z"
}
```

If you want to include attachments of articles, the payload looks like:

Request:

```
POST /api/v1/ticket_articles

{
   "ticket_id": 3,
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
   "content_type": "text/html",
   "type": "note",
   "internal": false,
   "time_unit": "12",
   "attachments": [
     {
       "filename": "some_file1.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     },
     {
       "filename": "some_file2.txt",
       "data": "content in base64",
       "mime-type": "text/plain"
     }
   ]
}
```

Response:

```
Status: 201 Created

{
   "id": 3,
   "from": "Bob Smith",
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
   "content_type": "text/html",
   "type": "note",
   "internal": false,
   "time_unit": "12.0"
   "attachments": [
     {
       "id": 123,
       "filename": "some_file1.txt",
       "preferences": {
         "Mime-Type": "text/plain"
       }
     },
```

(continues on next page)

(continued from previous page)

```
    {
      "id": 124,
      "filename": "some_file2.txt",
      "preferences": {
        "Mime-Type": "text/plain"
      }
    }
  ],
  ...
  "created_at": "2016-10-19T10:07:12.011Z",
  "updated_at": "2017-01-18T12:45:53.420Z"
}
```

To download attachments you need to call "GET /api/v1/ticket_attachment/#{ticket_id}/#{article_id}/#{id}".

If you want to add inline images, just use data URIs in HTML markup:

Request:

```
POST /api/v1/ticket_articles

{
  "ticket_id": 3,
  "to": "",
  "cc": "",
  "subject": "some subject",
  "body": "<b>some</b> message witn inline image <img src=\"
↪ABCDEFG==\">"
  "content_type": "text/html",
  "type": "note",
  "internal": false,
  "time_unit": "12"
}
```

Response:

```
Status: 201 Created

{
  "id": 3,
  "ticket_id": 3,
  "from": "Bob Smith",
  "to": "",
  "cc": "",
  "subject": "some subject",
  "body": "huhuhuu<br>huhuhuu<br>huhuhuu<br><br>",
  "content_type": "text/html",
  "type": "note",
  "internal": false,
  "time_unit": "12.0"
  "attachments": [
    {
      "id": 123,
      "filename": "44.262871107@zammad.example.com",
      "preferences": {
        "Mime-Type": "image/jpeg",
        "Content-ID": "44.262871107@zammad.example.com",
```

(continues on next page)

(continued from previous page)

```
          "Content-Disposition": "inline"
        }
      }
   ],
   ...
   "created_at": "2016-10-19T10:07:12.011Z",
   "updated_at": "2017-01-18T12:45:53.420Z"
}
```

To download attachments you need to call "GET /api/v1/ticket_attachment/#{ticket_id}/#{article_id}/#{id}".

If you want to create a phone ticket on behalf for a specific customer, use origin_by_id:

Required permission:

- ticket.agent (access to related ticket)

Request:

```
POST /api/v1/ticket_articles

{
   "ticket_id": 3,
   "origin_by_id": 5,
   "to": "",
   "cc": "",
   "subject": "some subject",
   "body": "<b>some</b> message witn inline image <img src=\"
↪ABCDEFG==\">"
   "content_type": "text/html",
   "sender": "Customer",
   "type": "phone",
   "internal": false,
   "time_unit": "12"
}
```

CHAPTER 40

Online Notification

40.1 List

Required permission:

- authenticated user (content of notitifcations depends on user permissions)

Request:

```
GET /api/v1/online_notifications
```

Response:

```
Status: 200 Ok

[
 {
   "id": 123,
   "o_id": 628,
   "object": "Ticket",
   "type": "escalation",
   "seen": true,
   "updated_at": "2016-08-16T07:55:42.119Z",
   "updated_by_id": 123,
   "created_at": "2016-08-16T07:55:42.119Z",
   "created_at_id": 123
 },
 {
   "id": 124,
   "o_id": 629,
   "object": "Ticket",
   "type": "update",
   "seen": false,
   "updated_at": "2016-08-16T07:55:47.119Z",
   "updated_by_id": 123,
```

(continues on next page)

(continued from previous page)

```
    "created_at": "2016-08-16T07:55:47.119Z",
    "created_at_id": 123
  },
  {
    "id": 125,
    "o_id": 630,
    "object": "Ticket",
    "type": "create",
    "seen": false,
    "updated_at": "2016-08-16T07:57:49.119Z",
    "updated_by_id": 123,
    "created_at": "2016-08-16T07:57:49.119Z",
    "created_at_id": 123
  },
]
```

40.2 Show

Required permission:

- authenticated user (content of notifications depends on user permissions)

Request:

```
GET /api/v1/online_notifications/{id}
```

Response:

```
Status: 200 Ok

{
  "id": 123,
  "o_id": 628,
  "object": "Ticket",
  "type": "escalation",
  "seen": true,
  "updated_at": "2016-08-16T07:55:42.119Z",
  "updated_by_id": 123,
  "created_at": "2016-08-16T07:55:42.119Z",
  "created_at_id": 123
}
```

40.3 Update

Required permission:

- admin.object

Request:

```
PUT /api/v1/online_notifications/{id}

{
```

(continues on next page)

(continued from previous page)

```
  "seen": true,
}
```

Response:

```
Status: 200 Ok

{
 "id": 123,
 "o_id": 628,
 "object": "Ticket",
 "type": "escalation",
 "seen": true,
 "updated_at": "2016-08-16T07:55:42.119Z",
 "updated_by_id": 123,
 "created_at": "2016-08-16T07:55:42.119Z",
 "created_at_id": 123
}
```

40.4 Delete

Required permission:

- authenticated user (content of notifications depends on user permissions)

Request:

```
DELETE /api/v1/online_notifications/{id}
```

Response:

```
Status: 200 Ok

{}
```

40.5 Mark all as read

Required permission:

- authenticated user (content of notifications depends on user permissions)

Request:

```
POST /api/v1/online_notifications/mark_all_as_read
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 41

Object

41.1 List

Required permission:

- admin (access to admin interface)

Request:

```
GET /api/v1/object_manager_attributes
```

Response:

```
Status: 200 Ok

[
   {
      "id":49,
      "name":"anrede",
      "display":"Anrede",
      "data_type":"select",
      "data_option":{
         "options":{
            "Mr":"Mr",
            "Ms":"Ms",
            "Company":"Company"
         },
         "default":"Mr",
         "null":true,
         "maxlength":255,
         "nulloption":true
      },
      "data_option_new":{

      },
```

(continues on next page)

(continued from previous page)

```
      "editable":true,
      "active":true,
      "screens":{
         "create":{
            "Customer":{
               "shown":true,
               "required":true
            }
         },
         "edit":{
            "Customer":{
               "shown":true
            },
            "Agent":{
               "shown":true
            }
         },
         "create_middle":{
            "Agent":{
               "shown":true
            }
         }
      },
      "to_create":false,
      "to_migrate":false,
      "to_delete":false,
      "to_config":false,
      "position":1550,
      "created_by_id":3,
      "updated_by_id":3,
      "created_at":"2017-01-13T16:19:23.116Z",
      "updated_at":"2017-01-17T11:16:13.298Z",
      "object":"Ticket"
   },
   # ...
]
```

41.2 Show

Required permission:

- admin (access to admin interface)

Request:

```
GET /api/v1/object_manager_attributes/:id
```

Response:

```
Status: 200 Ok

{
   "id":49,
   "name":"anrede",
   "display":"Anrede",
```

(continues on next page)

(continued from previous page)


```
   "data_type":"select",
   "data_option":{
      "options":{
         "Mr":"Mr",
         "Ms":"Ms",
         "Company":"Company"
      },
      "default":"Mr",
      "null":true,
      "maxlength":255,
      "nulloption":true
   },
   "data_option_new":{

   },
   "editable":true,
   "active":true,
   "screens":{
      "create":{
         "Customer":{
            "shown":true,
            "required":true
         }
      },
      "edit":{
         "Customer":{
            "shown":true
         },
         "Agent":{
            "shown":true
         }
      },
      "create_middle":{
         "Agent":{
            "shown":true
         }
      }
   },
   "to_create":false,
   "to_migrate":false,
   "to_delete":false,
   "to_config":false,
   "position":1550,
   "created_by_id":3,
   "updated_by_id":3,
   "created_at":"2017-01-13T16:19:23.116Z",
   "updated_at":"2017-01-17T11:16:13.298Z",
   "object":"Ticket"
}
```


41.3 Create

Required permission:

- admin (access to admin interface)

Request:

```
POST /api/v1/object_manager_attributes
```

Response:

```
Status: 200 Ok

{
   "name":"product",
   "object":"Ticket",
   "display":"Produkt",
   "active":true,
   "data_type":"select",
   "data_option":{
      "options":{
         "wert1":"anzeige1",
         "wert2":"anzeige12"
      }
   },
   "screens":{
      "create_middle":{
         "Customer":{
            "shown":true,
            "item_class":"column"
         },
         "Agent":{
            "shown":true,
            "item_class":"column"
         }
      },
      "edit":{
         "Customer":{
            "shown":true
         },
         "Agent":{
            "shown":true
         }
      }
   }
}
```

41.4 Update

Required permission:

- admin (access to admin interface)

Request:

```
PUT /api/v1/object_manager_attributes/:id
```

Response:

```
Status: 200 Ok
```

(continues on next page)

(continued from previous page)


```
{
   "id":49,
   "name":"anrede",
   "display":"Anrede",
   "data_type":"select",
   "data_option":{
      "options":{
         "Mr":"Mr",
         "Ms":"Ms",
         "Company":"Company"
      },
      "default":"Mr",
      "null":true,
      "maxlength":255,
      "nulloption":true
   },
   "data_option_new":{

   },
   "editable":true,
   "active":true,
   "screens":{
      "create":{
         "Customer":{
            "shown":true,
            "required":true
         }
      },
      "edit":{
         "Customer":{
            "shown":true
         },
         "Agent":{
            "shown":true
         }
      },
      "create_middle":{
         "Agent":{
            "shown":true
         }
      }
   },
   "to_create":false,
   "to_migrate":false,
   "to_delete":false,
   "to_config":false,
   "position":1550,
   "created_by_id":3,
   "updated_by_id":3,
   "created_at":"2017-01-13T16:19:23.116Z",
   "updated_at":"2017-01-17T11:16:13.298Z",
   "object":"Ticket"
}
```

41.5 Execute Database Migrations

Required permission:

- admin (access to admin interface)

Request:

```
POST /api/v1/object_manager_attributes_execute_migrations
```

Response:

```
Status: 200 Ok

{ }
```

CHAPTER 42

Tags

42.1 List

Required permission:

- ticket.agent or admin.tag

Request:

```
GET /api/v1/tags?object=Ticket&o_id=10
```

Response:

```
Status: 200 Ok

{
    "tags": [
        "tag 1",
        "tag 2",
        "tag 3"
    ]
}
```

42.2 Search

Required permission:

- ticket.agent or admin.tag

Request:

```
GET /api/v1/tag_search?term=tag
```

Response:

```
Status: 200 Ok

[
    {
        "id": 7,
        "value": "tag 1"
    },
    {
        "id": 8,
        "value": "tag 2"
    },
    {
        "id": 9,
        "value": "tag 3"
    }
]
```

42.3 Add

Required permission:

- ticket.agent or admin.tag

Request:

```
GET /api/v1/tags/add?object=Ticket&o_id=10&item=tag+4
```

Response:

```
Status: 200 Ok

true
```

42.4 Remove

Required permission:

- ticket.agent or admin.tag

Request:

```
GET /api/v1/tags/remove?object=Ticket&o_id=10&item=tag+4
```

Response:

```
Status: 200 Ok

true
```

42.5 Admin - List

Required permission:

- admin.tag

Request:

```
GET /api/v1/tag_list
```

Response:

```
Status: 200 Ok

[
    {
        "id": 7,
        "name": "tag 1",
        "count": 1
    },
    {
        "id": 8,
        "name": "tag 2",
        "count": 1
    },
    {
        "id": 9,
        "name": "tag 3",
        "count": 1
    },
    {
        "id": 11,
        "name": "tag 4",
        "count": 0
    },
    {
        "id": 6,
        "name": "test",
        "count": 0
    }
]
```

42.6 Admin - Create

Required permission:

- admin.tag

Request:

```
POST /api/v1/tag_list

{
  name: "tag 5"
}
```

Response:

```
Status: 200 Ok

{}
```

42.7 Admin - Rename

Required permission:

- admin.tag

Request:

```
PUT /api/v1/tag_list

{
  id: 6,
  name: "tag 5"
}
```

Response:

```
Status: 200 Ok

{}
```

42.8 Admin - Delete

Required permission:

- admin.tag

Request:

```
DELETE /api/v1/tag_list

{
  id: 6,
}
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 43

User Access Token

43.1 List

Required permission:

- user_preferences.access_token

Request:

```
GET /api/v1/user_access_token
```

Response:

```
Status: 200 Ok

{
  "tokens":[
    {"id":1,"label":"some user access token","preferences":{"permission":["cti.agent",
↪"ticket.agent"]},"last_used_at":null,"expires_at":null,"created_at":"2018-07-
↪11T08:18:56.947Z"}
    {"id":2,"label":"some user access token 2","preferences":{"permission":[ticket.
↪agent"]},"last_used_at":null,"expires_at":null,"created_at":"2018-07-11T08:18:56.
↪947Z"}
  ],
  "permissions":[
    {id: 1, name: "admin", note: "Admin Interface", preferences: {}, active: true,...}
↪,
    {id: 2, name: "admin.user", note: "Manage Users", preferences: {}, active: true,..
↪.},
    ...
  ]
 }
```

43.2 Create

Required permission:

- user_preferences.access_token

Request:

```
POST /api/v1/user_access_token

{
  "label":"some test",
  "permission":["cti.agent","ticket.agent"],
  "expires_at":null
}
```

Response:

```
Status: 200 Ok

{
  "name":"new_token_only_shown_once"
}
```

43.3 Delete

Required permission:

- user_preferences.access_token

Request:

```
PUT /api/v1/user_access_token/:id
```

Response:

```
Status: 200 Ok

{}
```

CHAPTER 44

Introduction

In many use cases, agents work in connection customer conversations over the phone.

It is a great relief when the telephone system (PBX) is integrated with Zammad, which makes processes with agents more effective.

The goal of the document is to provide the necessary API documentation to enable PBX vendors to easily integrate with Zammad.

44.1 Feature list

Inbound

- Caller identification based on the CallerID (open a customer profile with just one click)
- Display of open and closed tickets of a customer in a special overview. This overview should also give the possibility to create a ticket for the given customer.
- Intelligent mapping of CallerIDs with direct (e.g. directly at the contact) and not direct (e.g. telephone numbers from the signature)
- Caller Journal (which calls have been made and which have been handled and which require a callback)
- Blocking of CallerIDs (already during the call) *
- Support to allow an agent to set a DND - like state *
- Overview of agents who currently handle a call

Outbound

- Direct dialling of the customer telephone number and indexing of the call *
- Set the outbound caller ID based on the line phone number (e. g. set sender caller id based on country of destination caller id) *
- if supported by the PBX/telephone system

CHAPTER 45

Push API

45.1 How it works

Events can be transferred in realtime from the telephone system to the Zammad CTI Push API (REST API) via a generic interface.

Depending on the event, Zammad offers various functions to quickly and easily identify callers and the corresponding tickets, for example, or to provide a caller log. Or to modify the incoming or outgoing call.

45.2 Endpoint

The endpoint of your Zammad CTI Push API looks like http://localhost:3000/api/v1/cti/:token and can be found in Zammad -> Admin -> Integrations -> CTI (generic) -> Endpoint

45.3 Events

Zammad supports the following three events (newCall, hangup and answer) in version 2.x.

Event: newCall

Attribute	Description
event	"newCall"
from	The calling number (e.g. "493055571600" or "anonymous")
to	The called number (e.g. "491711234567890")
direction	The direction of the call (either "in" or "out")
callId	A unique alphanumeric identifier to match events to specific calls (max. 250 characters)
user[]	The user(s) realname involved. It is the name of the calling user when direction is "out", or of the users receiving the call when direction is "in". Group calls may be received by multiple users. In that case a "user[]" parameter is set for each of these users. It is always "user[]" (not "user"), even if only one user is involved.
queue	The queue name (e. g. helpdesk). This field is optional.

You can simulate this POST request and test your server with a CURL command:

```
curl -X POST --data "event=newCall&from=493055571600&to=491711234567890&direction=in&
↪callId=123456&user[]=Alice&user[]=Bob" http://localhost:3000/api/v1/cti/:token
```

The response (optional)

After sending the POST request to Zammad, your PBX can accept an JSON response to determine what to do (e. g. for *direction=in* to block the caller or for *direction=out* to set a caller id).

Zammad currently supports the following responses for incoming calls:

Action	Description
reject	Reject call or pretend to be busy (depending on your settings in Zammad)

Example 1: Reject call signaling busy

```
{
  "action": "reject",
  "reason": "busy"
}
```

Zammad currently supports the following responses for outgoing calls:

Action	Description
dial	To set the caller id (depending on your settings in Zammad). Number need to be in E.164 format.

Example 1: Set custom caller id for outgoing call

```
{
  "action": "dial",
  "callerId": "493055571642",
  "number": "491711234567890"
}
```

Event: hangup

Attribute	Description
event	“hangup”
callId	Same as in newCall-event for a specific call
cause	The cause for the hangup event (see
from	The calling number (e.g. “493055571600” or “anonymous”)
to	The called number (e.g. “491711234567890”)
direction	The direction of the call (either “in” or “out”)
answeringNumber	The number which was answering

You can simulate this POST request and test your server with a CURL command:

```
curl -X POST --data "event=hangup&cause=normalClearing&callId=123456&
↪from=493055571600&to=491711234567890&direction=in&answeringNumber=4921199999999"␣
↪http://localhost:3000/api/v1/cti/:token
```

Hangup causes: For these reasons, hangups may occur because of these causes:

Attribute	Description
normalClearing	One of the parties hung up after the call was established.
busy	The called party was busy
cancel	The caller hung up before the called party picked up
noAnswer	The called party rejected the call (e.g. through a DND setting)
congestion	The called party could not be reached
notFound	The called number does not exist or called party is offline
forwarded	The call was forwarded to a different party

Event: answer

Attribute	Description
event	“answer”
callId	Same as in newCall-event for a specific call
user	Name of the user who answered this call. Only incoming calls can have this parameter
from	The calling number (e.g. “492111234567” or “anonymous”)
to	The called number (e.g. “491711234567890”)
direction	The direction of the call (either “in” or “out”)
answeringNumber	The number of the answering destination. Useful when redirecting to multiple destinations

You can simulate this POST request and test your server with a CURL command:

```
curl -X POST --data "event=answer&callId=123456&user=John+Doe&from=493055571600&
↪to=491711234567890&direction=in&answeringNumber=21199999999" http://localhost:3000/
↪api/v1/cti/:token
```

CHAPTER 46

Backup and Restore

Zammad contains simple backup & restore scripts that can be executed via command line or cron job. You can find the scripts in the "/opt/zammad/contrib/backup" directory.

46.1 Configuration

- Rename "/opt/zammad/contrib/backup/config.dist" to "/opt/zammad/contrib/backup/config"
- Configure backup path in /opt/zammad/contrib/backup/config if you want. Standard backup path is "/var/tmp/zammad_backup"

46.2 Create Backup

```
cd /opt/zammad/contrib/backup
./zammad_backup.sh
```

46.3 Restore everything

```
cd /opt/zammad/contrib/backup
```

46.3.1 With menu for choosing backup date

```
./zammad_restore.sh
```

46.3.2 With command line argument for backup date

```
./zammad_restore.sh 20170507121848
```

CHAPTER 47

Configure environment variables

If you're using the DEB or RPM packages you can change Zammads environment variables by the following commands.

47.1 Configure IP

```
zammad config:set ZAMMAD_BIND_IP=0.0.0.0
systemctl restart zammad
```

47.2 Configure ports

Please note that you also have to reconfigure Nginx when changing the ports!

```
zammad config:set ZAMMAD_RAILS_PORT=3000
zammad config:set ZAMMAD_WEBSOCKET_PORT=6042
systemctl restart zammad
```

47.3 Application Servers

Per default one application server will get started. If you have more http requests (user sessions) you need to increase the amount of your application server. The typical problem is long waiting times in the web interface for opening or editing tickets.

```
zammad config:set WEB_CONCURRENCY=3
systemctl restart zammad
```

47.4 Configure Restart Command

If you need to make changes (creating objects) to Zammad, it can be necessary to restart the service. This can be done manually or automatic. If you like to use the automatic way you need to set an special environment variable.

Note: you might need to adjust the value for APP_RESTART_CMD if you have / need a different command to restart your Zammad on your installation.

```
zammad config:set APP_RESTART_CMD="systemctl restart zammad"
```

CHAPTER 48

Package Repo Files

Recently (24 Jul 2017) our packaging service provider (packager.io: http://www.packager.io/) improved its package distribution which makes it necessary that you update your Zammad repo file (e. g. /etc/apt/sources.list.d/zammad.list or /etc/yum.repos.d/zammad.repo) on your operating system.

If you're using an old repo file, you will not be able to update Zammad.

For more background information see: https://blog.packager.io/posts/24-change-of-repository-urls

Please use the following commands to update your Zammad repo file:

48.1 CentOS 7

```
sudo yum -y install epel-release wget
sudo wget -O /etc/yum.repos.d/zammad.repo https://dl.packager.io/srv/zammad/zammad/
↪stable/installer/el/7.repo
sudo yum install zammad
```

48.2 Debian 8

```
sudo apt-get install wget
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list https://dl.packager.io/srv/zammad/
↪zammad/stable/installer/debian/8.repo
sudo apt-get update
sudo apt-get install zammad
```

48.3 Debian 9

```
sudo apt-get install wget
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list https://dl.packager.io/srv/zammad/
↪zammad/stable/installer/debian/9.repo
sudo apt-get update
sudo apt-get install zammad
```

48.4 Ubuntu 16.04

```
sudo apt-get install wget
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list https://dl.packager.io/srv/zammad/
↪zammad/stable/installer/ubuntu/16.04.repo
sudo apt-get update
sudo apt-get install zammad
```

48.5 Ubuntu 18.04

```
sudo apt-get install wget
wget -qO- https://dl.packager.io/srv/zammad/zammad/key | sudo apt-key add -
sudo wget -O /etc/apt/sources.list.d/zammad.list \
  https://dl.packager.io/srv/zammad/zammad/stable/installer/ubuntu/18.04.repo
sudo apt-get update
sudo apt-get install zammad
```

48.6 SLES 12

```
sudo zypper install wget
sudo wget -O /etc/zypp/repos.d/zammad.repo https://dl.packager.io/srv/zammad/zammad/
↪stable/installer/sles/12.repo
sudo zypper install zammad
```

48.7 Note

If you're using an old repo file, you will get error messages like these:

```
E: Failed to fetch https://deb.packager.io/gh/zammad/zammad/dists/xenial/stable/
↪binary-amd64/Packages  Writing more data than expected (7831 > 1153)
E: Some index files failed to download. They have been ignored, or old ones used␣
↪instead.
```

```
Paket zammad-1.5.0-1500965473.2be861e2.centos7.x86_64.rpm not signed
```

CHAPTER 49

Privacy

49.1 What information is stored exactly on images.zammad.com, and for how long?

- We use images.zammad.com to serve user avatars (based on email address fetched from e. g. gravatar) and organization logo (based on domain used for Zammad login page after initial admin account creation).
  - images.zammad.com is only a proxy for images (e. g. found on public resources like gravatar)
  - md5 sums of email addresses are used to cache images for 7 days
  - md5 sums of domains are used to cache images for 30 days

49.2 Which other parts of the system do send data?

- Zammad use 4 online services
- you can enabled/disable all of them via Admin → System → Services

Note: You can also create and set up your own backends/services for this, if you want.

- Image: To serve user avatars (based on email address fetched from e. g. gravatar) and organization logo (based on domain used for Zammad login page after initial admin account creation).
  - images.zammad.com is only a proxy for images
  - md5 sums of email addresses are used to cache images for 7 days
  - md5 sums of domains are used to cache images for 30 days
- GeoCalendar: Zammad can handle SLAs, for SLAs calendars (time zone, working hours and vacation days are important). This GeoCalendar service is executed after initial admin account creation to automatically configure the calendar of the admin (time zone, vacation days, . . .).
  - GeoCalendar - No information is stored or cached on geo.zammad.com

- GeoIp: Zammad has a security feature to track user sessions based on the user's browser and country. So if your session or password is used (maybe stolen) on a new browser or from a different country, Zammad will inform the Agent about the new use of password/session via email.
  - GeoIp - No information is stored or cached on geo.zammad.com
- GeoLocation: A ticket overview will not only be shown in a regular table, but on a map. For this map we need to know the geo location of certain items.
  - GeoLocation - Currently there is only a google maps backend to lookup geo locations
- The source code of this services is available at:
  - https://github.com/zammad/zammad/tree/develop/lib/service