CKAN Installation

Required Time: ~ 5 hours.

FS Operating System: Ubuntu16.04.5 LTS (GNU/Linux 4.4.0-138-generic x86_64) English version

BS Operating System: Ubuntu16.04.5 LTS (GNU/Linux 4.4.0-138-generic x86_64) English version

CKAN Version: 2.8

**Last Update: ** February 2020

Glossary:

FS: The Frontend Server(s). The server or VM that holds the CKAN and Drupal instances. in High Availability the frontend servers could be more than one.

BS: The Backend Server. The server or VM that holds the required databases (PostgreSQL, MySQL, RedisDB, Solr, Virtouso...)

Install the required packages

1. Install the required packages in the FS:

```
sudo apt-get update
sudo apt-get install python-dev python-pip python-virtualenv git-core
```

Installed packages:

Package	Description
Python	The Python programming language, v2.7
pip	A tool for installing and managing Python packages
virtualenv	The virtual Python environment builder
Git	A distributed version control system

2. Install the required packages in the BS

```
sudo apt-get update
sudo apt-get install postgresql libpq-dev solr-jetty openjdk-8-jdk redis-server git-core
```

Installed packages:

Package	Description
PostgreSQL	The PostgreSQL database system, v9.3 or newer
libpq	The C programmer's interface to PostgreSQL
Git	A distributed version control system
Apache Solr	A search platform
Jetty	An HTTP server (used for Solr).
OpenJDK JDK	The Java Development Kit (used by Jetty)
Redis	An in-memory data structure store

Install CKAN into a Python virtual environment

Inside the FS:

1. Create a Python virtual environment (virtualenv) to install CKAN into, and activate it:

```
sudo mkdir -p /usr/lib/ckan/default
sudo chown `whoami` /usr/lib/ckan/default
virtualenv --no-site-packages /usr/lib/ckan/default
. /usr/lib/ckan/default/bin/activate
```

The final command above activates your virtualenv. The virtualenv has to remain active for the rest of the installation and deployment process, or commands will fail. You can tell when the virtualenv is active because its name appears in front of your shell prompt, something like this:

```
(default) $ _
```

2. Install the recommended `setuptools` version and up-to-date pip:

```
pip install setuptools==36.1
```

3. Install the CKAN source code into your virtualenv.

To install the latest stable release of CKAN (CKAN 2.8.2), run:

```
pip install -e 'git+https://github.com/ckan/ckan.git@ckan-2.8.2#egg=ckan'
```

4. Install the Python modules that CKAN requires into your virtualenv:

```
pip install -r /usr/lib/ckan/default/src/ckan/requirements.txt
```

Deactivate and reactivate your virtualenv, to make sure you're using the virtualenv's copies of commands like `paster` rather than any system-wide installed copies:

```
deactivate
. /usr/lib/ckan/default/bin/activate
```

Setup a PostgreSQL database:

Inside the BS:

```
sudo service postgresql start
```

Check that PostgreSQL was installed correctly by listing the existing databases:

```
sudo -u postgres psql -l
```

Press `CTRL+Z` to exit

Check that the encoding of databases is `UTF8`, if not internationalization may be a problem. Since changing the encoding of PostgreSQL may mean deleting existing databases, it is suggested that this is fixed before continuing with the CKAN install.

Create a new PostgreSQL database user called ckan_default, and enter a password for the user when prompted. You'll need this password later:

```
sudo -u postgres createuser -S -D -R -P ckan_default
```

Create a new PostgreSQL database, called ckan_default, owned by the database user you just created:

```
sudo -u postgres createdb -O ckan_default ckan_default -E utf-8
```

Now we need to make PostgreSQL listen from its public port. that's why we need to edit the `postgresql.conf`

Open to the postgresql config file

```
sudo nano /etc/postgresql/9.3/main/postgresql.conf
```

Go to line of `listen_addresses` parameter, uncomment it if it is commented, then replace the value with a '*' to make the postgresql listen to all ports

```
listen_addresses = '*'
```

If you would like to specify the ports, write a comma-separated list of IP addresses of the network interfaces PostgreSQL should listen, for example:

```
listen_addresses = 'localhost,192.168.1.21'
```

We also need to edit the pg_hba.conf to allow the machine running ckan to connect to PostgreSQL:

Open the pg_hba.conf:

```
sudo nano /etc/postgresql/9.3/main/pg_hba.conf
```

Add a line similar to the line below to the bottom of pg_hba.conf to allow the machine running Apache to connect to PostgreSQL. Please change the IP address as desired according to your network settings.

```
# TYPE  DATABASE       USER            ADDRESS                 METHOD
host  all       all       192.168.1.22/32         md5
```

if you have more than one ckan machines, you need to add the IP address of both machines example:

```
# TYPE  DATABASE       USER            ADDRESS                 METHOD
host  all       all       10.0.0.12/32         md5
host  all       all       10.0.0.24/32         md5
```

Create a CKAN config file

Inside the FS:

1. Create a directory to contain the site's config files:

```
sudo mkdir -p /etc/ckan/default
sudo chown -R `whoami` /etc/ckan/
```

2. Create the CKAN config file:

```
paster --plugin=ckan generate config /etc/ckan/default/production.ini
```

Edit the `production.ini` file in a text editor, changing the following options:

`sqlalchemy.url`

This should refer to the database we created above:

```
sqlalchemy.url = postgresql://ckan_default:pass@localhost/ckan_default
```

Replace `pass` with the password that you created

If you're using a remote host with password authentication rather than SSL authentication, use:

```
sqlalchemy.url = postgresql://ckan_default:pass@<remotehost>/ckan_default?sslmode=disable
```

`site_id`

Each CKAN site should have a unique `site_id`, for example:

```
ckan.site_id = default
```

`site_url`

Provide the site's URL (used when putting links to the site into the FileStore, notification emails etc). For example:

```
ckan.site_url = http://www.opendata.com
```

Do not add a trailing slash to the URL.

Setup Solr

CKAN uses Solr as its search platform, and uses a customized Solr schema file that takes into account CKAN's specific search needs. Now that we have CKAN installed, we need to install and configure Solr.

Inside the BS:

Edit the Jetty configuration file (`/etc/default/jetty8`) and change the following variables:

```
NO_START=0            # (line 4)
JETTY_HOST=127.0.0.1  # (line 16)
JETTY_PORT=8983       # (line 19)
```

change `JETTY_HOST` to the public IP of BS.

Start or restart the Jetty server.

```
sudo service jetty8 restart
```

You can test Solr responds correctly like this (you may need to install curl first):

```
$ curl http://localhost:8983/solr/

<html>
<head>
<link rel="stylesheet" type="text/css" href="solr-admin.css">
<link rel="icon" href="favicon.ico" type="image/ico"></link>
<link rel="shortcut icon" href="favicon.ico" type="image/ico"></link>
<title>Welcome to Solr</title>
</head>

<body>
<h1>Welcome to Solr!</h1>
<a href="."><img border="0" align="right" height="78" width="142" src="admin/solr_small.png" alt="Solr"/></a>


<a href="admin/">Solr Admin</a>


</body>
</html>
```

Replace the default `schema.xml` file with a symlink to the CKAN schema file included that we have in inside the manual in the CKAN installation directory.

```
cd /etc/solr/conf
sudo mv /etc/solr/conf/schema.xml /etc/solr/conf/schema.xml.bak
sudo nano schema.xml
```

Copy the text inside the schema.xml that is inside the manual in the CKAN installation directory. and paste it inside the created schema.xml (C+SHIFT+V) . to exit and save press (CTRL+X) and enter

restart Solr:

```
sudo service jetty8 restart
```

Inside the FS:

Change the `solr_url` setting in your (/etc/ckan/default/production.ini) to point to your Solr server, for example:

```
solr_url=http://10.0.2.14:8983/solr
```

Link to `who.ini`

Inside the FS:

`who.ini` (the Repoze.who configuration file) needs to be accessible in the same directory as your CKAN config file, so create a symlink to it:

```
ln -s /usr/lib/ckan/default/src/ckan/who.ini /etc/ckan/default/who.ini
```

Create database tables

```
. /usr/lib/ckan/default/bin/activate
cd /usr/lib/ckan/default/src/ckan
paster db init -c /etc/ckan/default/production.ini
```

You should see `Initialising DB: SUCCESS`.

Deploying CKAN

Install Apache, modwsgi, modrpaf

```
sudo apt-get install apache2 libapache2-mod-wsgi libapache2-mod-rpaf
```

Install NGINX

```
sudo apt-get install nginx
```

Install Email Server

```
sudo apt-get install postfix
```

Create your site's WSGI script file `/etc/ckan/default/apache.wsgi`

and paste the following content inside:

```
import os
activate_this = os.path.join('/usr/lib/ckan/default/bin/activate_this.py')
execfile(activate_this, dict(__file__=activate_this))

from paste.deploy import loadapp
config_filepath = os.path.join(os.path.dirname(os.path.abspath(__file__)), 'production.ini')
from paste.script.util.logging_config import fileConfig
fileConfig(config_filepath)
application = loadapp('config:%s' % config_filepath)
```

Create the Apache Config File

```
sudo nano /etc/apache2/sites-available/ckan_default.conf
```

and paste the following content inside:

```
WSGISocketPrefix /var/run/wsgi
<VirtualHost 0.0.0.0:8080>

    ServerName default.ckanhosted.com
    ServerAlias www.default.ckanhosted.com
    WSGIScriptAlias / /etc/ckan/default/apache.wsgi
    Alias /resource_cache "/var/lib/ckan/default/archiver"

    # pass authorization info on (needed for rest api)
    WSGIPassAuthorization On

    # Deploy as a daemon (avoids conflicts between CKAN instances)
    WSGIDaemonProcess ckan_default display-name=ckan_default processes=2 threads=15

    WSGIProcessGroup ckan_default

    ErrorLog /var/log/apache2/ckan_default.error.log
    CustomLog /var/log/apache2/ckan_default.custom.log combined

    <Directory />
    Require all granted
    </Directory>

    <Directory "/var/lib/ckan/default/archiver">
    Options +Indexes
    AllowOverride None
    Order allow,deny
    Allow from all
    </Directory>

</VirtualHost>
```

edit the `/etc/apache2/ports.conf` by changing `Listen 80` to `Listen 8080`

Create the NGINX config file in `/etc/nginx/sites-available/ckan`

```
sudo nano /etc/nginx/sites-available/ckan
```

Paste the content:

```
proxy_cache_path /tmp/nginx_cache levels=1:2 keys_zone=cache:30m max_size=250m;
proxy_temp_path /tmp/nginx_proxy 1 2;

server {
    client_max_body_size 100M;
    location / {
        proxy_pass http://127.0.0.1:8080/;
        proxy_set_header X-Forwarded-For $remote_addr;
        proxy_set_header Host $host;
        proxy_cache cache;
        proxy_cache_bypass $cookie_auth_tkt;
        proxy_no_cache $cookie_auth_tkt;
        proxy_cache_valid 30m;
        proxy_cache_key $host$scheme$proxy_host$request_uri;
        # In emergency comment out line to force caching
        # proxy_ignore_headers X-Accel-Expires Expires Cache-Control;
    }

}
```

Enable the CKAN site

```
 sudo a2ensite ckan_default
sudo a2dissite 000-default
sudo rm -vi /etc/nginx/sites-enabled/default
sudo ln -s /etc/nginx/sites-available/ckan /etc/nginx/sites-enabled/ckan_default
sudo service apache2 reload
sudo service nginx reload
```

You should now be able to visit your server in a web browser and see your new CKAN instance.