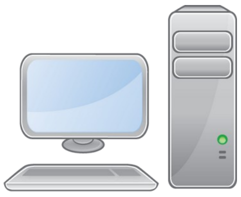
NOOSDrift



How to deploy a Central component

Table des matières

1. Introduction

This document is focused on the deployment and setting up of a Central component of the NOOS-Drift network.

2. Prerequisites

A linux system with

a) A console access

b) An installed Apache web server (the python module of this web server will be added in this document)

c) An installed PostgreSQL server with an empty database called « noosdrift » belonging to the user « noosdrift ». The database must be accessible by the localhost using the psql utility.

d) An installed FTP server with an available user for the Nodes and the Central or a direct access from the Central application to the files in the FTP directory.

e) Components “supervisor” and “rabbitmq-server” will also be installed.

3. Python 3.x Installation

This procedure makes sure that the right version of python is available (>=3,6). Further python modules will be installed automatically during deployment of the NOOS-Drift application code

a) On Ubuntu 14.04 LTS (Trusty Tahr) [with super user rights]

```
add-apt-repository ppa:jonathonf/python-3.6
apt update
apt install python3.6 python3-pip
pip3 install virtualenv
```

b) On Debian 8 (Jessie) [with super user rights]

```
apt install make build-essential libssl-dev zlib1g-dev
apt install libreadline-dev libsqlite3-dev wget curl llvm
apt install libncurses5-dev libncursesw5-dev xz-utils tk-dev
apt install libbz2-dev liblzma-dev libgdbm-dev checkinstall
wget https://www.python.org/ftp/python/3.6.3/Python-3.6.3.tgz
tar xvf Python-3.6.8.tgz
cd Python-3.6.8/
./configure --enable-optimizations --enable-shared –with-ensurepip=install ↲
            --without-gcc LDFLAGS="-Wl,--rpath=\$$LIB:/usr/local/lib"
make -j8
```

! Keep an eye on the output !
"Python build finished successfully!" is not enough!

```
checkinstall -D --install=no --fstrans=no --pkgname=python3.6 ↲
                  make altinstall
dpkg -i python3.6_3.6.8-1_amd64.deb
mv python3.6_3.6.8-1_amd64.deb /home/www-data/
cd
rm -r Python-3.6.8/
rm Python-3.6.8.tgz
```

c) On Debian 10 (Buster) [with super user rights]

Python 3 used by Debian Buster is Python 3.7

```
apt install python3-venv
```

d) On CentOS 7 [with super user rights]

```
yum install https://centos7.iuscommunity.org/ius-release.rpm
yum update
yum install python36u python36u-libs python36u-devel python36u-pip
```

4. Create python virtual environments

The configuration path and python virtual environments have been put in slightly different locations for different OSes.

/var/opt/noosdrift/
on Debian/Ubuntu type systems
/opt/noosdrift/
on CentOS systems

Both pathes will be refered to as /pathconfdir

a) On Ubuntu < 18.04 / Debian <= 8.0 [with super user rights]

```
cd /pathconfdir
mkdir -p Django
mkdir -p Django_env_vars
sudo apt install virtualenv
virtualenv -p /usr/bin/python3.6 ~/venv
```

b) On Debian 10, Ubuntu >= 18.04 and CentOS 7 [with super user rights]

```
cd /pathconfdir
mkdir -p Django
mkdir -p Django_env_vars
sudo python3 -m venv ./venv
```

5. Install apache mod_wsgi

The mod_wsgi module is required by apache so that it can run python applications. Debian and Ubuntu distributions now have well structured packages which make these procedures much simpler for Debian > 8.0 and Ubuntu > 14.04

a) On Ubuntu 14.04 LTS (Trusty Tahr) [with super user rights]

```
apt install python3.6-dev apache2-dev
su - www-data
source /pathconfdir/venv/Apache_WSGI_MOD_Env/bin/activate
pip install --no-cache-dir mod_wsgi
deactivate
exit
vim /etc/apache2/mods-available/wsgi.load
```

```
LoadModule wsgi_module
"/pathconfdir/venv/Apache_WSGI_MOD_Env/lib/python3.6/↲
site-packages/mod_wsgi/server/mod_wsgi-py36.cpython-36m-x86_64-linux-gnu.so"
WSGIPythonHome "/pathconfdir/venv/Apache_WSGI_MOD_Env/"
```

```
chown root:root /etc/apache2/mods-available/wsgi.load
chmod 644 /etc/apache2/mods-available/wsgi.load
a2enmod wsgi
service apache2 restart
service apache2 status                               (check)
apachectl -M                                         (check)
```

b) On Debian 8 (Jessie) [with super user rights]

```
apt install apache2-dev
su - www-data
source /pathconfdir/venv/Apache_WSGI_MOD_Env/bin/activate
pip install --no-cache-dir mod_wsgi
deactivate
exit
vim /etc/apache2/mods-available/wsgi.load
```

```
LoadModule wsgi_module
"/pathconfdir/venv/Apache_WSGI_MOD_Env/lib/python3.6/↲
site-packages/mod_wsgi/server/mod_wsgi-py36.cpython-36m-x86_64-linux-gnu.so"
WSGIPythonHome "/pathconfdir/venv/Apache_WSGI_MOD_Env/"
```

```
chown root:root /etc/apache2/mods-available/wsgi.load
chmod 644 /etc/apache2/mods-available/wsgi.load
a2enmod wsgi
service apache2 restart
service apache2 status                                (check)
apachectl -M                                          (check)
```

c) On Debian 10 (Buster) [with super user rights]

```
apt install libapache2-mod-wsgi-py3
```

d) On CentOS 7 [with super user rights]

```
yum install python36u-mod_wsgi
httpd -M                                              (check)
```

6. Set environment variables

Environment variables are actually set in 4 places because our code is started up for 4 different tasks and sharing variables between those systems is not possible.
Those systems are :

- The System Shell
- The Apache Application Server
- The Celery deamon
- The Background-task deamon

For each of these systems a specific file will be used to define and set the following list of variables necessary to the application.

`NOOS_BROKER_URL`	A url that will be used to access the rabbitMQ, responsible for proper order of processing of each demand
`NOOS_CENTRAL_ID`	A numeric reference to the central system
`NOOS_DBHOST`	The Database server host (localhost)
`NOOS_DBNAME`	The database name (noosdrift)
`NOOS_DBPORT`	The database server port (5432)
`NOOS_DBPWD`	The database user password (TheDBPassword)
`NOOS_DBUSER`	The database user for the application (noosdrift)
`NOOS_ENV`	An indication of whether the system is running in production or development mode (important for logging and error messages)
`NOOS_MME_ID`	A numeric reference to the system performing the MME analysis
`NOOS_NODE_ID`	A numeric reference to the present system
`NOOS_ROLE`	Is the present system a partner Node or the Central system
`NOOS_SECRET_KEY`	The secret key used encrypt the authentication procedure
`NOOS_USERNAME`	Access to the central system uses this user
`NOOS_USERPWD`	Access to the central uses this password

a) The System Shell.

Setting environment variables for this system requires modifying the contents of file :

`/pathconfdir/venv/bin/activate`

In the file, add towards the end of "deactivate" function:

```
#Unset NOOS_Drift variables
unset NOOS_BROKER_URL
unset NOOS_CENTRAL_ID
unset NOOS_DBHOST
unset NOOS_DBNAME
unset NOOS_DBPORT
unset NOOS_DBPWD
unset NOOS_DBUSER
unset NOOS_ENV
unset NOOS_MME_ID
unset NOOS_NODE_ID
unset NOOS_ROLE
unset NOOS_SECRET_KEY
unset NOOS_USERNAME
unset NOOS_USERPWD
```

Add at the end of file, replacing # with the correct values

```
#NOOS_Drift variables
NOOS_BROKER_URL="ampq://rabbitMQUser:rabbitMQPwd@localhost:5672/
rabbitMq_vhost"
NOOS_CENTRAL_ID="1"
NOOS_DBHOST="localhost"
NOOS_DBNAME="noosdrift"
NOOS_DBPORT="5432"
NOOS_DBPWD="ThePassword"
NOOS_DBUSER="noosdrift"
NOOS_ENV="PROC"
NOOS_MME_ID="1"
NOOS_NODE_ID="1"
NOOS_ROLE="Central"
NOOS_USERNAME="#"
NOOS_USERPWD="#"

export NOOS_BROKER_URL NOOS_CENTRAL_ID NOOS_DBHOST, NOOS_DBNAME,
NOOS_DBPORT, NOOS_DBPWD, NOOS_DBUSER NOOS_ENV NOOS_MME_ID NOOS_NODE_ID
NOOS_ROLE NOOS_SECRET_KEY  NOOS_USERNAME NOOS_USERPWD
```

b) The Apache Application Server [with super user rights]

This part set environment variables when the NOOS-Drift application is started by the Apache Server.

Setting the environement variables at this level requires to modify file :

`/pathconfdir/Django_env_vars/noosdrift.py`

Make sure the file exists, the you have the rights to edit it.

In the file, replace # with correct value

```
"""
Environment variables for noosDrift project.
Only strings are permitted.

"""

import os

os.environ["NOOS_BROKER_URL"] =
"ampq://rabbitMQUser:rabbitMQPwd@localhost:5672/rabbitMq_vhost"
os.environ["NOOS_CENTRAL_ID"] = "1"
os.environ["NOOS_DBHOST"]="localhost"
os.environ["NOOS_DBNAME"]="noosdrift"
os.environ["NOOS_DBPORT"]="5432"
os.environ["NOOS_DBPWD"]="ThePassword"
os.environ["NOOS_DBUSER"]="noosdrift"
os.environ["NOOS_ENV"] = "PROD"
os.environ["NOOS_MME_ID"] = "1"
os.environ["NOOS_NODE_ID"] = "#"
os.environ["NOOS_ROLE"] = "Node"
os.environ["NOOS_SECRET_KEY"] = "#"
os.environ["NOOS_USERNAME"] = "#"
os.environ["NOOS_USERPWD"] = "#"
```

```
chown root:root /pathconfdir/Django_env_vars/noosdrift.py
chmod 644 /pathconfdir/Django_env_vars/noosdrift.py
```

c) The Celery deamon

See Celery configuration for this (Chapter 10)

d) The Background-task deamon

See Background task configuration for this (Chapter 11)

7. Deploy application code

   If the git software is not yet installed. Install it using apt (Debian/Ubuntu) or yum (CentOS)

   a) Clone the code from a git repository

```
cd  /pathtconfdir
source venv/bin/activate

git clone -b centralbranch --single-branch ↲
git@github.com:rbins-swap-team/NoosDrift.git Django/noosDrift

pip install --no-cache-dir --upgrade --force-reinstall -r requirements.txt
python manage.py migrate
python manage.py collectstatic --settings=deploy_settings
vim noosDrift/settings.py
```

Check or edit "ALLOWED_HOST"

```
deactivate
cd ../..
```

   b) Set the appropriate user rights for this directory (and under)

   On Ubuntu 14.04 LTS (Trusty Tahr), Debian 8 (Jessie), 10 (Buster)

```
chown -R www-data.www-data Django
```

   On CentOS 7

```
chown -R apache.apache Django
```

8. Install RabbitMQ

   a) On Ubuntu 14.04 LTS (Trusty Tahr), Debian 8 (Jessie), Debian 10 (Buster) [with super user rights]

```
apt install rabbitmq-server
rabbitmqctl add_user noosdrift theRabbitmqPwd
rabbitmqctl add_vhost noosdrift_vhost
rabbitmqctl set_permissions -p noosdrift_vhost noosdrift ".*" ".*" ".*"
rabbitmqctl delete_user guest
```

Change value of NOOS_BROKER_URL parameter to

```
ampq://noosdrift:theRabbitmqPwd@localhost:5672:noosdrift_vhost
```

in files

```
/pathconfdir/venv/bin/activate
/pathconfdir/Django_env_vars/noosdrift.py
```

Modify credentials for Celery to match what was made at step 2 & 3

b) On CentOS 7 [super user]

```
sudo yum install rabbitmq-server
sudo systemctl enable rabbitmq-server.service
sudo systemctl start rabbitmq-server.service
sudo rabbitmqctl add_user noosdrift theRabbitmqPwd
sudo rabbitmqctl add_vhost noosdrift_vhost
sudo rabbitmqctl set_premissions -p noosdrift_vhost noosdrift ".*" ".*" ".*"
sudo rabbitmqctl delete_user guest
```

Change value of NOOS_BROKER_URL parameter to

```
ampq://noosdrift:theRabbitmqPwd@localhost:5672:noosdrift_vhost
```

in files

```
/pathconfdir/venv/bin/activate
/pathconfdir/Django_env_vars/noosdrift.py
```

Modify credentials for Celery to match what was made at step 4 & 5

9. Check Celery

a) On Ubuntu 14.04 LTS (Trusty Tahr), Debian 8 (Jessie), Debian 10 (Buster) [www-data]

```
cd /pathconfdir/
source venv/bin/activate
cd Django/noosDrift
clear && celery -A noosDrift worker -E --loglevel=INFO
[Ctrl + C]
[Ctrl + C]
deactivate
```

b) On CentOS 7 [with super user rights]

```
cd /pathconfdir/
source venv/bin/activate
cd Django/noosDrift
clear && celery -A noosDrift worker -E --loglevel=INFO
[Ctrl + C]
[Ctrl + C]
deactivate
```

10. Daemonizing Celery using Supervisord (system package version) [with super user rights]

```
apt install supervisor
vi /etc/supervisor/supervisord.conf
```

Add into [supervisord] section

```
user=root
```

Add into both [unix_http_server] and [supervisorctl] sections

```
username=noos#dummy_user
password=noos#dummy_password
```

```
mkdir /var/log/celeryd
touch /var/log/celeryd/noosdrift.log
touch /var/log/celeryd/noosdrift_error.log
mkdir /etc/supervisor/conf.d
vi /etc/supervisor/conf.d/noosdrift-celery.conf
```

Replace # with correct value

```
[program:noosdrift-celery]
command=/pathconfdir/venv/bin/celery -A noosDrift worker –loglevel=info
environment=PYTHONPATH="/pathconfdir/Django/noosDrift",↲
            PATH="/pathconfdir/venv/bin:%(ENV_PATH)s",↲
NOOS_BROKER_URL=”ampq://
            noosdrift:theRabbitmqPwd@localhost:5672:noosdrift_vhost”,↲
            NOOS_CENTRAL_ID="1",NOOS_DBHOST="localhost", NOOS_DBNAME="noosdrift",↲
            NOOS_DBPORT="5432", NOOS_DBPWD="#", NOOS_DBUSER="noosdrift", ↲
            NOOS_ENV="PROD",NOOS_MME_ID="1",NOOS_NODE_ID="#",NOOS_ROLE="Central",↲
            NOOS_SECRET_KEY="#",NOOS_USERNAME="#",NOOS_USERPWD="#"
directory=/pathconfdir/Django/noosDrift
user=www-data
numprocs=1
stdout_logfile=/var/log/celeryd/noosdrift.log
stderr_logfile=/var/log/celeryd/noosdrift_error.log
autostart=true
autorestart=true
startsecs=10
stopwaitsecs=600
priority=998
```

```
chmod 640 /etc/supervisor/conf.d/noosdrift-celery.conf
systemctl restart supervisor
```

Check those logging files to see if any error occurs :

```
/var/log/celeryd/noosdrift.log
/var/log/celeryd/noosdrift_error.log
/var/log/supervisor/supervisord.log
```

11. Deamonizing « background-tasks » using supervisor

    We add file noosdrift-background-tasks.conf to directory /etc/supervisor/conf.d/

```
[program:noosdrift-background-tasks]
command=/var/opt/noosdrift/Django/noosDrift/manage.py process_tasks
environment=PYTHONPATH="/var/opt/noosdrift/Django/noosDrift",↲
PATH="/var/opt/noosdrift/venv/bin:%(ENV_PATH)s",↲
NOOS_ENV="PROD",NOOS_ROLE="Central",NOOS_MME_ID=1,NOOS_NODE_ID=1,↲
NOOS_CENTRAL_ID=1,NOOS_DBHOST="localhost",NOOS_DBNAME="noosdrift",↲
NOOS_DBPORT=5432,NOOS_DBUSER="noosdrift",NOOS_DBPWD="NOOS-Drift",↲
NOOS_SSH_MME="",NOOS_USERNAME="#",NOOS_USERPWD="#",NOOS_BROKER_URL=↲
"amqp://noosdrift:noosdrift=celery_2019@localhost:5672/noosdrift_vhost",↲
NOOS_SECRET_KEY="#",NOOS_SFTP_HOST="ftp.rbins.be",↲
NOOS_SFTP_USER="noosdrift_sftp",NOOS_SFTP_PWD="#"

directory=/var/opt/noosdrift/Django/noosDrift
user=www-data
numprocs=1
stdout_logfile=/var/log/supervisor/noosdrift-background-tasks.log
stderr_logfile=/var/log/supervisor/noosdrift-background-tasks_error.log
autostart=true
autorestart=true
startsecs=10
stopwaitsecs = 600
priority=998
```

12. Using Django with Apache2 [with super user rights]

    Add to the appropriate config file (eg. /etc/apache2/sites-available/default-ssl.conf)

```
##########
# Django #
##########

#NOOS-Drift
Alias /noosdrift/api/static "/pathconfdir/Django/noosDrift/static"
Alias /noosdrift/api/media "/pathconfdir/Django/noosDrift/media"
Alias /noosdrift/api/templates "/pathconfdir/Django/noosDrift/templates"

<Directory "/pathconfdir/Django/noosDrift/static">
        Options Indexes FollowSymLinks
        AllowOverride None
        Require all granted
</Directory>
<Directory "/pathconfdir/Django/noosDrift/media">
        Options -Indexes
        AllowOverride None
        Require all granted
</Directory>
<Directory "/pathconfdir/Django/noosDrift/templates">
        Options -Indexes
        AllowOverride None
        Require all granted
</Directory>
<Directory "/pathconfdir/Django/noosDrift/noosDrift">
        <Files "wsgi.py">
                Require all granted
        </Files>
</Directory>

WSGIScriptAlias /noosdrift/api /pathconfdir/Django/noosDrift/noosDrift/wsgi.py
WSGIDaemonProcess noosDriftTest user=www-data group=www-data ↲
                  python-home=/pathconfdir/venv ↲
                  python-path=/pathconfdir/venv/lib/python3.X/site-packages/: ↲
                  /pathconfdir/Django/noosDrift
WSGIPassAuthorization On
WSGIProcessGroup noosDriftTest

WSGIApplicationGroup %{GLOBAL}
```

```
a2ensite default-ssl                    (if site not enabled yet)
systemctl restart apache2        (if site not enabled yet)
systemctl reload apache2         (if site already enabled)
```

Check if url `noosdrift/api` is available.

13. The FTP server

An FTP server must be installed with a proper user to transfert files between NOOS-Drift components.
The directory used by the ftpuser must therefore be accessible to the application.
www-data or apache must be part of the group and the group must have read and write access rights to the files in this directory.
It is in this directory that the central application is going to look for partner node results.
This ftp directory is mounted on directory
`/pathtoconfig/Django/noosdrift/requests`
But this can be accomplished just as well using a symbolic link.
The ftpuser and password must be known by the Node software but since we do not use it directly on the central software, it is not necessary to keep these settings there

14. How to check server processes are active

Command
```
ss -tlp
```
should give back a list of active protocols among which :

- http (the apache server is active)
- amqp (the rabbitmq server is active)
- postgresql (the postgresql server is active)
- ftp (the ftp server is active)

15. How to check the Node application is active

Simple tests allow to check if the application is active.

- From the local machine allows to see if the site is active

```
wget http://localhost/noosdrift/api/admin
```

- From an external machine

```
wget https://nodemachine/api/schema/core.json
```

If these command get's a normal response. Then the site is ready to work, although further settings might be required (passwords, accounts, transfer settings).
If the site does not answer then something is very wrong and it would be better to examin the deployment in depth

16. Node component finer settings

Besides adding values to the environment parameters in files « activate », « noosdrift.py » « noosdrift-celery.conf », some values need to be set in the « settings.py » file in the Django app.

a) ALLOWED HOSTS (Already mentionned)
b) Settings linked to security certificates. This varies from one system to another. Some systems have the certificate on the app machine, some not.
c) schema_dict

17. Important Central system files and directories

   a) Logging (What’s going on)

```
/var/log/celery/noosdrift.log
/var/log/celery/noosdrift_error.log
/var/log/supervisor/supervisor.log
/var/opt/noosdrift/Django/noosdrift/logs/
```

   b) Configuration (The grains of sand that blocks the whole system)

```
/pathconfdir/venv/bin/activate
/pathconfdir/Django/noosDrift/noosDrift/settings.py
/pathconfdir/Django_env_vars/noosdrift.py
/etc/supervisor/conf.d/noosdrift-celery.conf
/etc/supervisor/conf.d/noosdrift-background-task.conf
```

   c) Central result files (How did the Model respond)

   On a the Central system, the archive files comming from the Nodes are stored here

```
/var/opt/noosdrift/Django/noosdrift/requests/
```

   As archive files with names of the form

```
noosdrift_NN_NN_NN.tgz
```

   These files are unpacked to :

```
/var/opt/noosdrift/Django/noosdrift/results/NN/
```

   Where NN is the number of the demand.

   !!!!! Make sure that directory !!!!!

```
/var/opt/noosdrift/Django/noosdrift/results/
```

   is readable, writeable and executable by www-data or apache user

18. Important remark about the MME software component.

   The point of the Central is to execute a piece of software referred to as ‘the MME Analysis’. This software will be executed by the Central using www-data / apache user rights. So make sure that these software components are set with the appropriate Unix access rights.

19. What to do if the Node goes down ??

   The activity of the Node typically depends on the Apache server which must be active. The Node application itself (which is accessed through Apache) can be restarted by forcing the timestamp of file using (as superuser)

```
touch pathconfig/Django/noosDrift/noosDrift/wsgi.py
```

   Doing this forces Apache to restart the Node application

20. Add new Node

   Any new Node must be listed on the Central system. This is done through the web interface by the system adminstrator.

21. Add / Delete a new User

   Adding a new user happens through a user demand and manager confirmation process. So does the deactivation / deletion of the user.

22. NOOS-Drift application updates
    1. Activate the virtual environment
    2. Update the application code using git
    3. Update python tools pip, setuptools wheels
    4. Update the python modules using pip and the requirements file
    5. Update the database structure
    6. Force Apache to restart of the NOOS-Drift application by « touching » wsgi.py