Setting up a NORDUni 3.0 instance

- Setting up NOCLook
  - NEO4J database
    - Install Java8
    - NEO4J
    - Configuration
  - Postgres database
  - NOCLook
    - Configure NOCLook
    - Migrate databases and check config
    - Create local superuser
    - Deploying NOCLook
      - UWSGI
      - NGINX
    - SAML SP
      - Local saml metadata
- Upgrade from NI 2 to 3
  - NEO4j 3.2
    - Debian package
    - Docker image
    - Migating data
      - Let neo4j handle the database migration
      - Import ni backup
    - Setup neo4j user password
  - Upgrade NI
    - Troubleshooting

Setting up NOCLook

NOCLook is the main GUI component, a django webapp that allows users to access the data stored in NI, it is also often what people refer to when they say NI.

This guide is written for Ubuntu 16.04.

NEO4J database

The official neo4j installation guide for version 3.2 is the reference for this part.

Install Java8

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
$ java -version
java version "1.8.x"
```

NEO4J

```
$ wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -
$ echo 'deb http://debian.neo4j.org/repo stable/' | sudo tee -a /etc/apt/sources.list.d/neo4j.list
$ sudo apt-get update
$ sudo apt-get install neo4j=3.2.2
```

Configuration

```
$ sudo vim /etc/neo4j/neo4j.conf
Add the following:

# Autoindexing
# Enable auto-indexing for nodes, default is false
node_auto_indexing=true

# The node property keys to be auto-indexed, if enabled
node_keys_indexable=name, description, ip_address, ip_addresses, as_number, hostname, hostnames,
telenor_tn1_number, nordunet_id, version

# Enable auto-indexing for relationships, default is false
relationship_auto_indexing=true

# The relationship property keys to be auto-indexed, if enabled
relationship_keys_indexable=ip_address
```

```
$ sudo rm /var/lib/neo4j/data/dbms/auth
# Note the extra space before the command to avoid saving password in bash history
$  sudo -u neo4j neo4j-admin set-initial-password your_awesome_password
$ sudo service neo4j restart
```

Postgres database

Set password for database user and create a new database

```
$ sudo apt-get install postgresql
$ sudo -u postgres psql postgres
template1=# CREATE USER ni with PASSWORD 'secret';
template1=# CREATE DATABASE norduni;
template1=# GRANT ALL PRIVILEGES ON DATABASE norduni to ni;
template1=# ALTER DATABASE norduni OWNER TO ni;             #  Allow user ni to drop and create for restoring
template1=# ALTER USER ni CREATEDB;                          #  and development purposes
template1=# \q
```

NOCLook

Before installing NOCLook you need to install the required system libraries

```
$ sudo apt-get install git python-pip libpq-dev
$ sudo pip install -U pip
$ sudo pip install virtualenv
$ sudo adduser --disabled-password --home /var/opt/norduni ni
```

Now you are ready to install NOCLook, start by changing to the ni user.

```
$ sudo -u ni -i
$ pwd
/var/opt/norduni
$ git clone git://git.nordu.net/norduni.git
# Create virtual env
$ virtualenv norduni_environment
# Activate virtual env
$ . norduni_environment/bin/activate
# Install python dependencies
$ pip install -r norduni/requirements/prod.txt
```

Configure NOCLook

```
$ cd /var/opt/norduni/norduni/src/niweb/
$ cp dotenv .env
$ vi .env
```

You need to setup the following settings:

```
NEO4J_USERNAME=neo4j
NEO4J_PASSWORD=
REPORTS_TO=
SECURITY_REPORTS_TO=
DB_PASSWORD=
ALLOWED_HOSTS=ni.yourdomain.tld localhost
DEFAULT_FROM_EMAIL=
EMAIL_HOST=
SECRET_KEY=
```

The secret key should be at least 50 chars long consisting of the following characters: 'abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*(-_=+)'

You can use the following snippit to gennerate such a string locally:

```
# Missing ! since bash is sad when you use ! in anything, and pyton thinks \! is to be read as both \ and !
$ python -c "import random; print(''.join([random.SystemRandom().choice
('abcdefghijklmnopqrstuvwxyz0123456789@#$%^&*(-_=+)') for i in range(50)]))"
```

Migrate databases and check config

```
# To make it easier for yourself set DJANGO_SETTINGS_MODULE=niweb.settings.prod in your bashprofile/bashrc
$ cd /var/opt/norduni/norduni/src/niweb
$ python manage.py migrate
$ python manage.py collectstatic
$ python manage.py runserver
$ rm -r /tmp/django_cache
```

Create local superuser

```
$ python manage.py createsuperuser
```

Deploying NOCLook

Start by installing uwsgi and nginx.

```
$ sudo apt-get install nginx-full uwsgi uwsgi-plugin-python
```

UWSGI

```
$ sudo vi /etc/uwsgi/apps-available/noclook.ini

The following configuration should be a good start.

[uwsgi]
# Django-related settings
plugins = python
protocol = uwsgi
# the base directory (full path)
chdir           = /var/opt/norduni/norduni/src/niweb/
# Django's wsgi file
wsgi-file       = /var/opt/norduni/norduni/src/niweb/niweb/wsgi.py
env             = DJANGO_SETTINGS_MODULE=niweb.settings.prod
# the virtualenv (full path)
home            = /var/opt/norduni/norduni_environment
# logging
daemonize       = /var/log/uwsgi/app/noclook.log
# process-related settings
# master
master          = true
# maximum number of worker processes
processes       = 5
#threads        = 2
max-requests    = 5000
# the socket (use the full path to be safe
socket          = 127.0.0.1:8001
# clear environment on exit
vacuum          = true
# for now we run uwsgi in lazy-apps, due to neo4j session problems
lazy-apps       = true
# less noisy uwsgi logs (especially with sentry)
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true
```

Link the configuration in to the correct directory.

```
$ sudo ln -s /etc/uwsgi/apps-available/noclook.ini /etc/uwsgi/apps-enabled/noclook.ini
```

Make temp dir and log dir writable by the uwsgi user (www-data on ubuntu)

```
sudo chown -R ni:www-data /tmp/django_cache
sudo chmod -R g+rw /tmp/django_cache

sudo chown -R ni:www-data /var/opt/norduni/norduni/src/niweb/logs/
sudo chmod -R g+w /var/opt/norduni/norduni/src/niweb/logs/
```

Finally restart uwsgi

```
$ sudo service uwsgi restart
```

NGINX

Setup new dhparam file 2048 should suffice, but if you like you can go with 4096 instead:

```
$ sudo openssl dhparam -out /etc/ssl/dhparams.pem 2048
```

Configure nginx.

```
$ sudo vi /etc/nginx/sites-available/default

# The following configuration should be a good start.
# Remember certificates or
# sudo openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/ni_nordu_net.key -
out /etc/ssl/certs/ni_nordu_net.crt

upstream django {
    server 127.0.0.1:8001; # for a web port socket
}

server {
    listen         80;
    listen         [::]:80;
    server_name    ni.nordu.net;
    return         301 https://$server_name$request_uri;
}

server {
    listen 443;
    listen [::]:443 default ipv6only=on; ## listen for ipv6
    ssl on;
    ssl_certificate /etc/ssl/certs/ni_nordu_net.crt;
    ssl_certificate_key /etc/ssl/private/ni_nordu_net.key;

    # https://cipherli.st
    ssl_prefer_server_ciphers on;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_ciphers "EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH";
    ssl_session_cache shared:SSL:10m;
    ssl_ecdh_curve secp384r1;
    ssl_dhparam /etc/ssl/dhparams.pem;

    server_name ni.nordu.net;

    location /static/ {
        alias         /var/opt/norduni/norduni/src/niweb/niweb/static/;
        autoindex    on;
        access_log   off;
        expires      30d;
    }

    location / {
        include     /etc/nginx/uwsgi_params;
        proxy_set_header   Host                 $host;
        proxy_set_header   X-Real-IP            $remote_addr;
        proxy_set_header   X-Forwarded-For      $proxy_add_x_forwarded_for;
        proxy_set_header   X-Forwarded-Proto    $scheme;
        proxy_redirect     off;
        uwsgi_pass  django;
    }
}
```

Then restart nginx (still remember to setup ssl certificates)

```
$ sudo service nginx restart
```

SAML SP

If you want to set up NOCLook as a SAML SP you need to install the following packages and Python modules.

```
$ sudo apt-get install libffi-dev xmlsec1
$ sudo -u ni -i
$ . norduni_environment/bin/activate
$ pip install djangosaml2
```

You then need to uncomment the lines in settings.py that imports and sets up djangosaml2. You also have to create a pysaml2 configuration.
All this is best described in the documentation at https://pypi.python.org/pypi/djangosaml2.

Local saml metadata

To speed up login you can use local metadata. This metadata still needs to be updated and verified, and for that you can use https://github.com/NORDUnet/metadata-updater

You need to configure djangosaml2 to use local metadata, and you will have to add the meta-dataupdater to cron, preferably by running crontab -e as the ni user. Once an hour is reasonable, once a day can be ok, once a week might be tiresome when the cert expires.

Upgrade from NI 2 to 3

NEO4j 3.2

You can choose to either install the debian package of neo4j 3.2 or use the docker version of neo4j.

```
# Stop the old neo4j installation, and remove it from rc.d
$ sudo service neo4j-service stop
$ sudo update-rc.d -f neo4j-service remove
```

Debian package

You can follow the installation steps from the setup guide above. The guide below for migrating assumes you don't do the password creation step, so if you do you can skip the set-defa

If you use the debian package, you might want to change your database path, default is `/var/lib/neo4j/data`.

Docker image

You can also choose to run NEO4j as a docker image.

1. Pull the relevant docker image
2. Make a data dir
3. Change the password by connecting to the database
4. Only expose the ports locally 127.0.0.1:7474:7474
5. Setup system.d or other way of keeping NEO4j running all the time.

Migating data

There are two ways of migrating data for ni, either let neo4j handle the database upgrade or restore ni data from backup.

Let neo4j handle the database migration

1. Stop neo4j service if running
2. Remove data/databases/graph.db in 3.2 `sudo rm -r /var/lib/neo4j/data/databases/graph.db/`
3. Enable data migration in neo4j.conf `dbms.allow_format_migration=true` (should probably be disabled after first migration is done)
4. Import database from 2.x `sudo -u neo4j neo4j-admin import --mode=database --database=graph.db --from=/var/opt/neo4j-community-2.1.8/data/graph.db/``
5. Start neo4j service `sudo service neo4j start`
    a. you can follow along in `logs/neo4j.log`

Import ni backup

1. First make sure you have a backup of your current ni data (use the src/scripts/noclook_producer.py -O some_dir along with dumping the postgresql database using pg_dump)
2. Use src/scripts/noclook_consume.py with a conf file that points to noclook data created with noclook_producer

Setup neo4j user password

After migrating your data you need to set a password on you neo4j database.

You might have to remove the old auth settings first: `sudo rm /var/lib/neo4j/data/dbms/auth`

```
# Note the extra space in front to not have the password in your bash history
$  sudo -u neo4j neo4j-admin set-initial-password your_awesome_password
```

Upgrade NI

```
# In norduni directory as the NI user
$ sudo -u ni -i
$ cd norduni
$ git stash
$ git pull origin master
$ git stash apply
# You might get conflict e.g. in urls.py
# Resolve it and run git reset
$ rm -r src/niweb/norduniclient

# Delete old virtualenv and create a new
$ cd ..
$ rm -r norduni_environment
$ virtualenv norduni_environment
$ . norduni_environment/bin/activate
$ pip install -U pip
$ pip install -U -r norduni/requirements/prod.txt
# If running python 2.7 also run
$ pip install -r norduni/requirements/py2.txt

# update norduni/src/niweb/.env to have:
        NEO4J_USERNAME=neo4j
    NEO4J_PASSWORD=your_awesome_password
    NEO4J_RESOURCE_URI=bolt://localhost:7687
# If you have saml enabled you need to add the following to norduni/src/niweb/apps/saml2auth/config.py
# Just beside the key_file and cert_file entries.
        'encryption_keypairs': [{
        'key_file': path.join(BASEDIR, 'sp-key.pem'),  # private part
        'cert_file': path.join(BASEDIR, 'sp-cert.pem'),  # public part
    }],

# finally run migrate and collect statics
$ python norduni/src/niweb/manage.py migrate --settings=niweb.settings.prod
$ python norduni/src/niweb/manage.py collectstatic --settings=niweb.settings.prod
```

Then you can restart uwsgi.

```
$ rm -r /tmp/django_cache
$ sudo service uwsgi restart
```

Troubleshooting

If you run into problems you can use `python src/niweb/manage.py -h` to see if there are errors.

- `AttributeError: 'NoneType' object has no attribute 'session'` is caused by wrong credentials (or missing configuration there of)
- `ImportError: cannot import name contextmanager` or `ImportError: cannot import name IntegrityError` - you need to delete `src/niweb /norduniclient`
- `ImportError: No module named django.core.exceptions` (you need to source your virtual environment)
- `ImportError: No module named neo4j.v1.exceptions` - Problems with virtualenv installed dependencies, delete your virtualenv and install again
- `neo4j.exceptions.AuthError: The client is unauthorized due to authentication failure.` - your password contains some characters that got mangled, e.g. # or @, remove
- `IOError: [Errno 13] Permission denied: '/tmp/django_cache/'` - the directory is probably owned by ni user, and not www-data, which uwsgi runs as. Remove the dir, and restart uwsgi