Netbox - IT Dokumentation

Netbox is a "IP address management (IPAM) and data center infrastructure management (DCIM) tool".

https://github.com/netbox-community/netbox

At Wikimedia it is used as the DCIM and IPAM system, as well as being used as an integration point for switch and port management, DNS management, and similar operations.

Installation

Installation auf Debian 11

Abbhängigkeiten installieren

1.
```
apt update
apt -y install -y git gcc nginx redis supervisor python3 python3-dev python3-pip python3-setuptools
build-essential libxml2-dev libxslt1-dev libffi-dev graphviz libpq-dev libssl-dev zlib1g-dev
```

2. Installieren und von Konfigurieren von PostgreSQL

```
apt update
apt -y install postgresql-contrib postgresql-13-ip4r
service postgresql start
```

Nun eine Datenbank und Benutzer für Netbox anlegen.
Strong Passwort mit eingenem Passwort erstezten

```
su postgres
psql

CREATE DATABASE netbox;
CREATE USER netbox WITH PASSWORD 'StrongPassword';
GRANT ALL PRIVILEGES ON DATABASE netbox TO netbox;
\q
exit
```

Überpfrüfen das wir uns auch einloggen können. Dann wird nach dem Kennwort gefragt

```
psql -U netbox -h localhost -W

Ausgabe:
psql (13.7 (Debian 13.7-0+deb11u1))
SSL-Verbindung (Protokoll: TLSv1.3, Verschlüsselungsmethode: TLS_AES_256_GCM_SHA384, Bits: 256,
Komprimierung: aus)
Geben Sie »help« für Hilfe ein.
```

```
mit \q wieder aus
```

3. Installieren und konfigurieren von netbox
   in das Verzsichnis opt wechsels und git clone von netbox

```
cd /opt/
git clone -b master https://github.com/digitalocean/netbox.git
```

Konfigurations Datei kopieren

```
cd netbox/netbox/netbox/
cp configuration_example.py configuration.py
```

Nun die Konfigurationdatei bearbeiten

```
nano configuration.py
....
ALLOWED_HOSTS = ['localhost']
....
DATABASE = {
    'NAME': 'netbox',             # Database name
    'USER': 'netbox',             # PostgreSQL username
    'PASSWORD': 'StrongPassword', # PostgreSQL password
    'HOST': 'localhost',          # Database server
    'PORT': '',                   # Database port (leave blank for default)
    'CONN_MAX_AGE': 300,          # Max database connection age
 }
```

Django Schlüssel erstellen der muss in die Netbox config eingetragen werden.

```
apt -y install python-is-python3
cd /opt/netbox/netbox
./generate_secret_key.py

Ausgabe Der Key:
L2lyoE^*DN)6w3PK_d$-pe5ZS@XmMQ4J9g!cvF1V=n0juWiATR
```

Nun wieder die Konfigurationsdatei öffnen

```
cd netbox/netbox/netbox/
nano configuration.py

am Ende in der Datei
...
SECRET_KEY = ''
in
SECRET_KEY = 'L2lyoE^*DN)6w3PK_d$-pe5ZS@XmMQ4J9g!cvF1V=n0juWiATR'
```

Nun Netbox Abbhängigkeiten installieren

```
pip3 install -r /opt/netbox/requirements.txt
```

Datenbank einspielen

```
cd /opt/netbox/netbox/
python3 manage.py migrate
```

Beispielausgabe der Einspielens

```
Operations to perform:
  Apply all migrations: admin, auth, circuits, contenttypes, dcim, extras, ipam, secrets, sessions, taggit, tenancy, users, virtualization
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying taggit.0001_initial... OK
```

```
  Applying taggit.0002_auto_20150616_2121... OK
  Applying tenancy.0001_initial_squashed_0005_change_logging... OK
  Applying dcim.0001_initial... OK
  Applying ipam.0001_initial... OK
  Applying dcim.0002_auto_20160622_1821... OK
  Applying extras.0001_initial_squashed_0013_objectchange... OK
  Applying ipam.0002_vrf_add_enforce_unique... OK
  Applying dcim.0003_auto_20160628_1721_squashed_0010_devicebay_installed_device_set_null... OK
  Applying ipam.0003_ipam_add_vlangroups_squashed_0011_rir_add_is_private... OK
  Applying dcim.0011_devicetype_part_number_squashed_0022_color_names_to_rgb... OK
  Applying ipam.0012_services_squashed_0018_remove_service_uniqueness_constraint... OK
  Applying dcim.0023_devicetype_comments_squashed_0043_device_component_name_lengths... OK
  Applying virtualization.0001_virtualization... OK
  Applying ipam.0019_virtualization_squashed_0020_ipaddress_add_role_carp... OK
  Applying dcim.0044_virtualization_squashed_0061_platform_napalm_args... OK
  Applying extras.0014_configcontexts_squashed_0019_tag_taggeditem... OK
  Applying dcim.0062_interface_mtu_squashed_0065_front_rear_ports... OK
  Applying circuits.0001_initial_squashed_0006_terminations... OK
  Applying dcim.0066_cables...
    Adding console connections... 0 cables created
    Adding power connections... 0 cables created
    Adding interface connections... 0 cables created
 OK
  Applying circuits.0007_circuit_add_description_squashed_0017_circuittype_description...
    Adding circuit terminations... 0 cables created
 OK
  Applying tenancy.0006_custom_tag_models... OK
  Applying virtualization.0002_virtualmachine_add_status_squashed_0009_custom_tag_models... OK
  Applying secrets.0001_initial_squashed_0006_custom_tag_models... OK
  Applying ipam.0021_vrf_ordering_squashed_0025_custom_tag_models... OK
  Applying dcim.0067_device_type_remove_qualifiers_squashed_0070_custom_tag_models... OK
  Applying extras.0020_tag_data_squashed_0021_add_color_comments_changelog_to_tag... OK
  Applying
dcim.0071_device_components_add_description_squashed_0088_powerfeed_available_power...
Updating cable device terminations...
 OK
  Applying dcim.0089_deterministic_ordering... OK
  Applying dcim.0090_cable_termination_models... OK
  Applying extras.0022_custom_links_squashed_0034_configcontext_tags... OK
```

```
Applying extras.0035_deterministic_ordering... OK
Applying extras.0036_contenttype_filters_to_q_objects... OK
Applying ipam.0026_prefix_ordering_vrf_nulls_first_squashed_0032_role_description... OK
Applying ipam.0033_deterministic_ordering... OK
Applying secrets.0007_secretrole_description... OK
Applying sessions.0001_initial... OK
Applying taggit.0003_taggeditem_add_unique_index... OK
Applying users.0001_api_tokens_squashed_0003_token_permissions... OK
Applying virtualization.0010_cluster_add_tenant_squashed_0012_vm_name_nonunique... OK
Applying virtualization.0013_deterministic_ordering... OK
```

Admin Benutzer für Netbox erstellen

```
python3 manage.py createsuperuser

Beispiel Ausgabe/Einagbe:

Username (leave blank to use 'root'): admin
Email address: admin@example.com
Password: <Enter Password>
Password (again): <Re-enter Password>
Superuser created successfully.
```

Feste Dateien verschieben von netbox

```
cd /opt/netbox/netbox
python3 manage.py collectstatic
```

4. Installieren und konfigurieren von gunicorn

```
pip3 install gunicorn
```

Konfigurieren gunicorn für netbox

```
In eins markieren kopieren und einfügen

cat <<EOF | tee /opt/netbox/gunicorn_config.py
command = '/usr/local/bin/gunicorn'
pythonpath = '/opt/netbox/netbox'
```

```
bind = 'localhost:8085'
workers = 3
user = 'www-data'
EOF
```

5. Configurieren von supervisord

Konfigurationsdatei erstellen

```
In eins markieren kopieren und einfügen

cat <<EOF | tee /etc/supervisor/conf.d/netbox.conf
[program:netbox]
command = gunicorn -c /opt/netbox/gunicorn_config.py netbox.wsgi
directory = /opt/netbox/netbox/
user = www-data
EOF
```

supervisord neustarten und beim start enablen

```
systemctl restart supervisor.service
systemctl enable supervisor.service
```

Im Status sollte Netbox mit aufgelistet sein

```
systemctl status  supervisor
```

```
Ausgabe:

supervisor.service - Supervisor process control system for UNIX
     Loaded: loaded (/lib/systemd/system/supervisor.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2022-07-09 11:20:27 CEST; 3min 29s ago
       Docs: http://supervisord.org
   Main PID: 12646 (supervisord)
      Tasks: 5 (limit: 2340)
     Memory: 264.0M
        CPU: 6.745s
     CGroup: /system.slice/supervisor.service
             ├─12646 /usr/bin/python3 /usr/bin/supervisord -n -c /etc/supervisor/supervisord.conf
             ├─12649 /usr/bin/python3 /usr/local/bin/gunicorn -c /opt/netbox/gunicorn_config.py
netbox.wsgi
```

```
        ├─12650 /usr/bin/python3 /usr/local/bin/gunicorn -c /opt/netbox/gunicorn_config.py
netbox.wsgi
        ├─12651 /usr/bin/python3 /usr/local/bin/gunicorn -c /opt/netbox/gunicorn_config.py
netbox.wsgi
        └─12652 /usr/bin/python3 /usr/local/bin/gunicorn -c /opt/netbox/gunicorn_config.py
netbox.wsgi

Jul 09 11:20:27 netbox systemd[1]: Started Supervisor process control system for UNIX.
Jul 09 11:20:27 netbox supervisord[12646]: 2022-07-09 11:20:27,349 CRIT Supervisor is running as
root.  Privileges we>
Jul 09 11:20:27 netbox supervisord[12646]: 2022-07-09 11:20:27,349 INFO Included extra file
"/etc/supervisor/conf.d/n>
Jul 09 11:20:27 netbox supervisord[12646]: 2022-07-09 11:20:27,353 INFO RPC interface 'supervisor'
initialized
Jul 09 11:20:27 netbox supervisord[12646]: 2022-07-09 11:20:27,353 CRIT Server 'unix_http_server'
running without any>
Jul 09 11:20:27 netbox supervisord[12646]: 2022-07-09 11:20:27,353 INFO supervisord started with
pid 12646
Jul 09 11:20:28 netbox supervisord[12646]: 2022-07-09 11:20:28,357 INFO spawned: 'netbox' with pid
12649
Jul 09 11:20:29 netbox supervisord[12646]: 2022-07-09 11:20:29,745 INFO success: netbox entered
RUNNING state, proces>
lines 1-23/23 (END)
```

6. Nginx Web Server konfigurieren.
   eine Neue nginx Seite erstellen. Den servernamen durch den eigentlichen namen ersetzen.
   ist es lokal einfach einen hostnamen wählen und diesen dann in dem Client unter der Hosts Datei eintragen, so das der Webbrowser diesen namen aufrufen kann. Oder wer einen eigen DNS Server diesen Namen dort propagieren.

```
nano /etc/nginx/conf.d/netbox.conf


Inhalt:


server {
    listen 80;
    server_name netbox.example.com;
    client_max_body_size 25m;
```

```
    location /static/ {
        alias /opt/netbox/netbox/static/;
    }

    location / {
        proxy_pass http://localhost:8085;
        # proxy_set_header X-Forwarded-Host $server_name;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header Host $host;
        # proxy_set_header X-Forwarded-Proto $scheme;
        proxy_pass_header X-XSRF-TOKEN;

    }
}
```

Syntax Check der config

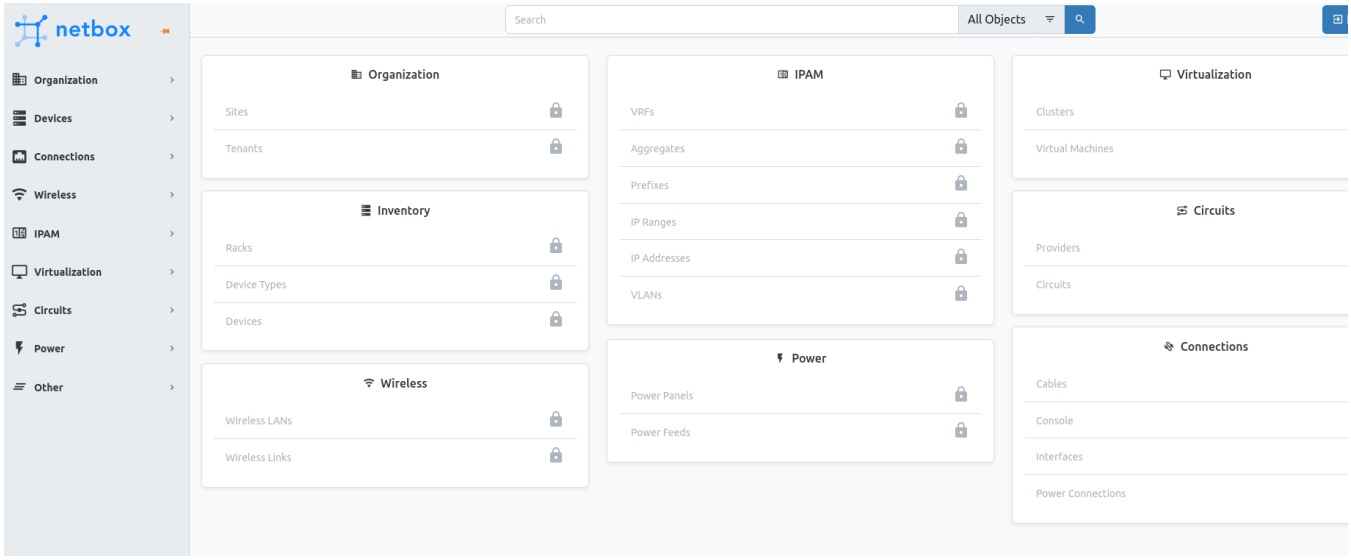
```
nginx -t

Ausgabe sollte sein:
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Nginx neustarten

```
systemctl restart nginx
```

7. Nun endlich anmelden an der Weboberfläche
   http://servername bei mir
   http://netbox.hacker.local.lan

   Diese domain ist auch in meister hosts Datei auf meinem Client Computer / Laptop



Einloggen. Fertig

Installation mit Docker

Beschreibung:

Für einige Dienste ist die Wartung mittelös Docker wesentlich einfacher. Neues images pullen, fertig.
Docker container reinstallieren.
Durch eine composer Datei nichts leichter als das.

Los gehts

Docker installieren:

siehe Buch hier klicken.

Datenverzeichnis erstellen

WIr legen unsere daten in /root/netboxdata ab. Und die configs für nginx im Unterverzeichnis nginx.

```
mkdir -p /root/netboxdata/nginx/
mkdir -p /root/netboxdata/netbox/media
mkdir -p /root/netboxdata/compose
```

Selbstsignierte SSL-Zertifikate erstellen

Wir ertsellen das Zertifikat mit 100 Jahren

```
openssl req -x509 -nodes -days 36500 -newkey rsa:2048 -keyout /root/netboxdata/nginx/selfsigned.key -out /root/netboxdata/nginx/selfsigned.crt
```

Nun die Fragen beantworten.
DE alles andere leer lassen außer

Common Name (e.g. server FQDN or YOUR name) []:netbox.local.lan

```
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:DE
State or Province Name (full name) [Some-State]:
Locality Name (eg, city) []:
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:netbox.local.lan
```

Docker-Compose-File erstellen

Die Compose Datei:

```
nano /root/netboxdata/compose/docker-compose.yml
```

Nun den Inhalt einfügen und ersetze `mysecretkey`, `mydbpassword` und `mydbrootpassword` mit sicheren, zufälligen Werten.
Der Secret KEY muss mindesten 50 Zeichen haben.

```
version: '3.1'

services:
  netbox:
    image: netboxcommunity/netbox
    depends_on:
      - mariadb
      - redis
    volumes:
      - /root/netboxdata/netbox/media:/opt/netbox/netbox/media
    environment:
      - NETBOX_SECRET_KEY=mysecretkey
```

```
      - NETBOX_ALLOWED_HOSTS=*
      - DB_NAME=netbox
      - DB_USER=netbox
      - DB_PASSWORD=mydbpassword
      - DB_HOST=mariadb
      - DB_DRIVER=mysql
      - REDIS_HOST=redis

  mariadb:
    image: mariadb:10.5
    environment:
      - MYSQL_ROOT_PASSWORD=mydbrootpassword
      - MYSQL_DATABASE=netbox
      - MYSQL_USER=netbox
      - MYSQL_PASSWORD=mydbpassword
    volumes:
      - /root/netboxdata/mariadb:/var/lib/mysql

  redis:
    image: redis:6

  nginx:
    image: nginx:1.21
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
      - /root/netboxdata/nginx:/etc/ssl/nginx:ro
    ports:
      - "80:80"
      - "443:443"
    depends_on:
      - netbox

networks:
  default:
    driver: bridge
```

NGINX Configuration erstellen

```
nano /root/netboxdata/compose/nginx.conf
```

Inhalt

```
worker_processes 1;

events {
    worker_connections 1024;
}

http {
    log_format main '$proxy_protocol_addr - $remote_user [$time_local] "$request" '
                    '$status $body_bytes_sent "$http_referer" '
                    '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;

    sendfile on;
    tcp_nopush on;
    tcp_nodelay on;
    keepalive_timeout 65;
    types_hash_max_size 2048;

    include /etc/nginx/mime.types;
    default_type application/octet-stream;

    server {
        listen 80;
        server_name _;
        return 301 https://$host$request_uri;
    }

    server {
        listen 443 ssl;
        server_name _;

        ssl_certificate /etc/ssl/nginx/selfsigned.crt;
        ssl_certificate_key /etc/ssl/nginx/selfsigned.key;

        location / {
            proxy_pass http://netbox:8001;
            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;
      }
   }
}
```

Docker-Compose-Setup

```
cd /root/netboxdata/compose/
docker-compose up -d
```

Datenbank initialiseren nach start des Containers

```
docker-compose run --rm netbox python3 manage.py migrate
```