How To Install NetBox IPAM on Debian 10

NetBox is an open source IPAM / DCIM web application used for managing and documenting computer networks and managing IP addresses. It was Initially conceived by the network engineering team at DigitalOcean. The tool is written in Django Python framework and relies on PostgreSQL database for data store.

NetBox building blocks:

- **IP address management (IPAM)** – IP networks and addresses, VRFs, and VLANs
- **Equipment racks –** Organized by group and site
- **Devices** – Types of devices and where they are installed
- **Connections** – Network, console, and power connections among devices
- **Virtualization** – Virtual machines and clusters
- **Data circuits** – Long-haul communications circuits and providers
- **Secrets** – Encrypted storage of sensitive credentials

If you’re interested in deploying Netbox on other systems, checkout:
Here are the steps for installing Install NetBox on Debian 10 (Buster) Linux.

Step 1: Install required dependencies

Start by installing all dependency applications required to run NetBox:

```
#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
```

Step 2: Install and configure PostgreSQL

NetBox uses PostgreSQL database server to store its data. So install PostgreSQL server on Debian 10:

```
#apt update
#apt -y install postgresql-contrib postgresql-11-ip4r
```

The Create a database and user for NetBox.

```
#su - postgress

$psql

CREATE DATABASE netbox;
CREATE USER netbox WITH PASSWORD 'Security@321';
GRANT ALL PRIVILEGES ON DATABASE netbox TO netbox;
\q
```

Confirm that you can login to database as `netbox` user.

```
#psql -U netbox -h localhost -W
Password:
psql (11.5 (Debian 11.5-1+deb10u1))
SSL connection (protocol: TLSv1.3, cipher:
TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.
netbox=> \q

#redis-cli ping
```

Step 3: Install and configure Netbox

Change to `/opt/` directory and clone project code.

```
#cd /opt/
```

```
# wget https://github.com/netbox-community/netbox/archive/refs/tags/v2.10.8.tar.gz

#ls

#tar vxf v2.10.8.tar.gz

#ls

#mv netbox*/ /opt/netbox

#ls
```

Create a configuration file from provided example file.

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

Edit the configuration file and set allowed host and database login details:

```
$  vim configuration.py
....
ALLOWED_HOSTS = ['localhost']
....
DATABASE = {
     'NAME': 'netbox',                  # Database name
     'USER': 'netbox',                  # PostgreSQL username
     'PASSWORD': 'Security@321', # PostgreSQL password
     'HOST': 'localhost',               # Database server
     'PORT': '',                        # Database port (leave 
blank for default)
     'CONN_MAX_AGE': 300,               # Max database 
connection age
 }
```

Generate Django SECRET Key:

```
#cd /opt/netbox/netbox
#./generate_secret_key.py
```

Then set the key on the file `/opt/netbox/netbox/netbox/configuration.py`

Example:

```
#nano /opt/netbox/netbox/netbox/configuration.py
SECRET_KEY = 'L2lyoE^*DN)6w3PK_d$-pe5ZS@XmMQ4J9g!cvF1V=n0juWiATR'
```

Install Netbox dependencies:

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

Migrate database data:

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

Sample output for database migration.

```
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_nul
l... 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
```

Create admin user:

```
#python3 manage.py createsuperuser

Username (leave blank to use 'root'): superadmin
Email address: superadmin@rashedacademy.com
Password: Security@321
Password (again): Security@321

Superuser created successfully.
```

Move static files

```
#cd /opt/netbox/netbox
#python3 manage.py collectstatic
280 static files copied to '/opt/netbox/netbox/static'.
```

Step 3: Install and configure gunicorn

Install gunicorn using pip3:

```
#pip3 install gunicorn
Collecting gunicorn
   Downloading
https://files.pythonhosted.org/packages/69/ca/926f7cd3a2014
b16870086b2d0fdc84a9e49473c68a8dff8b57f7c156f43/gunicorn-
20.0.4-py2.py3-none-any.whl (77kB)
     100% |████████████████████████████████| 81kB 1.7MB/s
 Requirement already satisfied: setuptools>=3.0 in
/usr/lib/python3/dist-packages (from gunicorn) (40.8.0)
 Installing collected packages: gunicorn
 Successfully installed gunicorn-20.0.4
```

Configure gunicorn for Netbox:

```
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
```

Step 4: Configure supervisord

Create a supervisord configuration file:

```
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
```

Restart and enable supervisord service to start on boot.

```
#systemctl restart supervisor.service
#systemctl enable supervisor.service
#systemctl status  supervisor
```

Step 5: Configure Nginx Web Server

Let’s configure Nginx web server to help us access Netbox via Domain name rather than specifying an IP address and a port.

Create new Nginx configuration file for Netbox.

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

With below data.

```
server {
    listen 80;
    server_name 103.36.103.19;
    client_max_body_size 25m;

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

    location / {
        proxy_pass http://localhost:8085;
    }
}
```

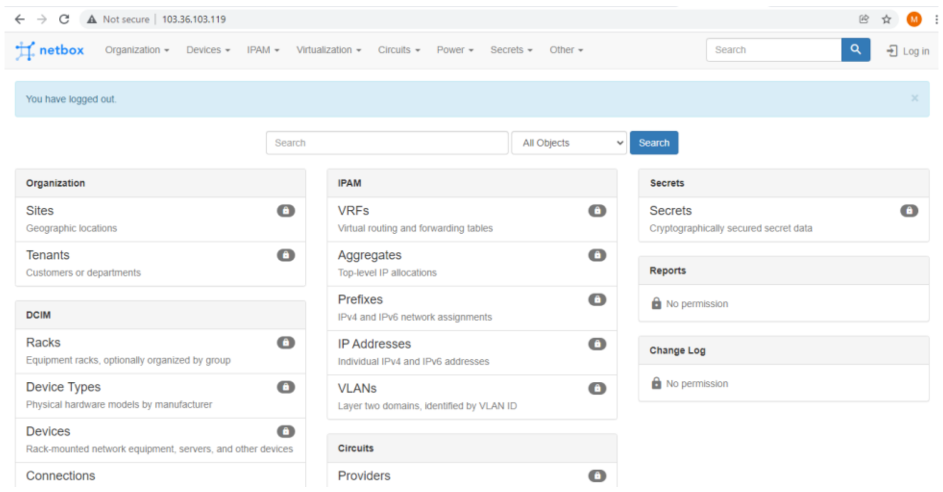
Check Nginx configuration syntax and restart its service

```
#nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax
is ok
nginx: configuration file /etc/nginx/nginx.conf test is
successful
```

If OK, restart Nginx service

```
#systemctl restart nginx
```

Step 6: Access Netbox Web UI



You have logged out.

Organization	
Sites — Geographic locations	
Tenants — Customers or departments	

DCIM	
Racks — Equipment racks, optionally organized by group	
Device Types — Physical hardware models by manufacturer	
Devices — Rack-mounted network equipment, servers, and other devices	
Connections	

IPAM	
VRFs — Virtual routing and forwarding tables	
Aggregates — Top-level IP allocations	
Prefixes — IPv4 and IPv6 network assignments	
IP Addresses — Individual IPv4 and IPv6 addresses	
VLANs — Layer two domains, identified by VLAN ID	

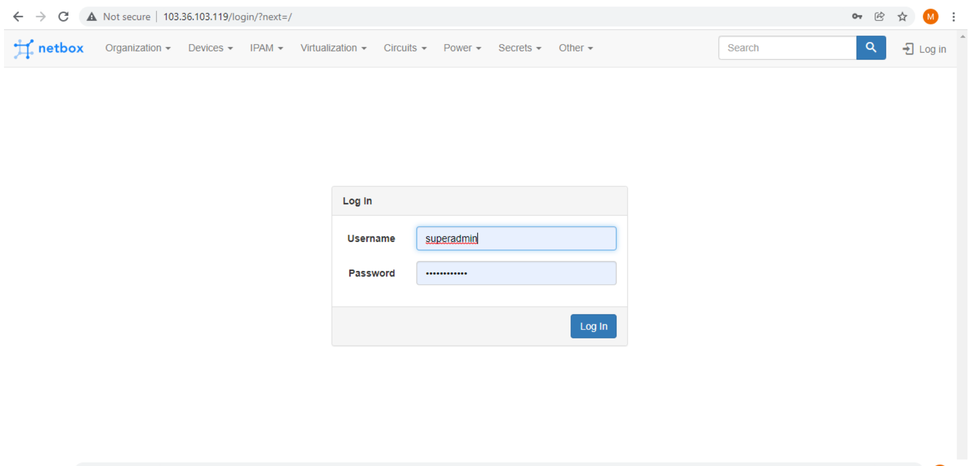
Circuits	
Providers	

Secrets	
Secrets — Cryptographically secured secret data	

Reports: No permission

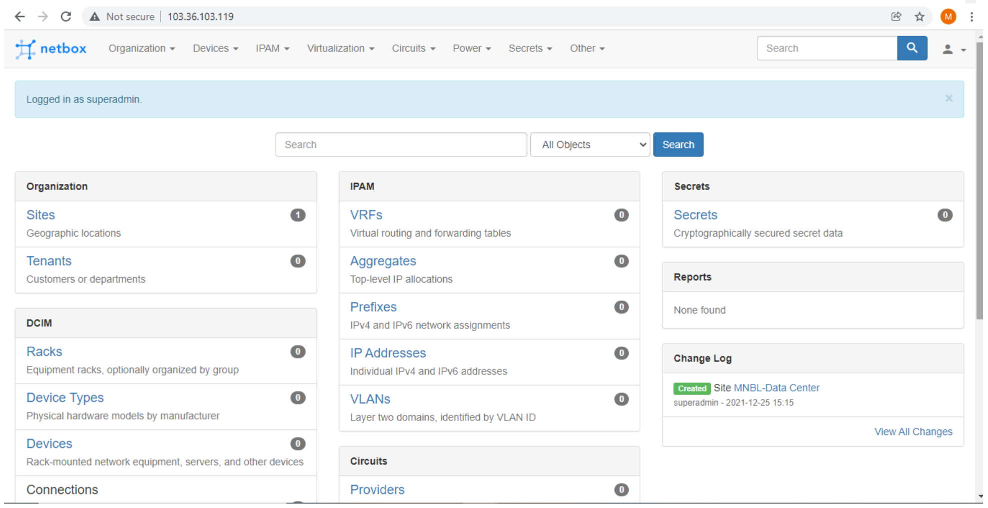
Change Log: No permission

Login into Netbox Using the following Username and Password:



Username: superadmin

Password: ••••••••••••



Logged in as superadmin.

Organization	
Sites — Geographic locations	1
Tenants — Customers or departments	0

DCIM	
Racks — Equipment racks, optionally organized by group	0
Device Types — Physical hardware models by manufacturer	0
Devices — Rack-mounted network equipment, servers, and other devices	0
Connections	

IPAM	
VRFs — Virtual routing and forwarding tables	0
Aggregates — Top-level IP allocations	0
Prefixes — IPv4 and IPv6 network assignments	0
IP Addresses — Individual IPv4 and IPv6 addresses	0
VLANs — Layer two domains, identified by VLAN ID	0

Circuits	
Providers	0

Secrets	
Secrets — Cryptographically secured secret data	0

Reports: None found

Change Log: Created Site MNBL-Data Center
superadmin - 2021-12-25 15:15
View All Changes

Netbox Dashboard