Docker Compose

- Netbox
- _Docker Daemon

Netbox

My docker-compose:

```
version: "3.7"
services:
  netbox-postgres:
    image: postgres:15-alpine
    container_name: netbox_postgres
    hostname: netbox-postgres
    restart: unless-stopped
    healthcheck:
      test: ["CMD", "pg_isready", "-q", "-d", "netbox", "-U", "netbox"]
      timeout: 45s
      interval: 10s
      retries: 10
    volumes:
      - /mnt/app_data/Server/Web-App/_docker-stack/netbox/postgres:/var/lib/postgresql/data
    env_file: stack.env
    environment:
      - POSTGRES_DB=netbox
      - POSTGRES_USER=netbox

  netbox-redis:
    image: redis:7-alpine
    user: 1026:100
    command: redis-server
    container_name: netbox_redis
    hostname: netbox-redis
    healthcheck:
      test: ["CMD-SHELL", "redis-cli ping || exit 1"]
    restart: always
    volumes:
      - /mnt/app_data/Server/Web-App/_docker-stack/netbox/redis:/data

  netbox-server:
    image: lscr.io/linuxserver/netbox:latest
```

```
  container_name: netbox_server
  hostname: netbox-server
  env_file: stack.env
  environment:
    - PUID=1026
    - PGID=100
    - TZ=America/Los_Angeles
    - ALLOWED_HOSTS=['netbox.domain.com', 'netbox.domain.internal', '172.122.122.122', '127.0.0.1']
    - DB_NAME=netbox
    - DB_USER=netbox
    - DB_HOST=netbox-postgres
    - DB_PORT=5432
    - REDIS_HOST=netbox-redis
    - REDIS_PORT=6379
    - REDIS_DB_TASK=0
    - REDIS_DB_CACHE=1
  volumes:
    - /mnt/app_data/Server/Web-App/_docker-stack/netbox/config:/config
  ports:
    - 13031:8000
  depends_on:
    - netbox-postgres
    - netbox-redis
  restart: unless-stopped
```

When I changed the IP addresses I expected to only need to change the hosts IP config in the compose however it was necessary here as below as well.

nano ~/netbox/config/configuration.py

```
#########################
#                       #
#   Required settings   #
#                       #
#########################

# This is a list of valid fully-qualified domain names (FQDNs) for the NetBox server. NetBox will not permit write
# access to the server via any other hostnames. The first FQDN in the list will be treated as the preferred name.
#
# Example: ALLOWED_HOSTS = ['netbox.example.com', 'netbox.internal.local']
ALLOWED_HOSTS = ['netbox.domain.com', 'netbox.domain.internal', '172.122.122.122', '127.0.0.1']
```

```
# PostgreSQL database configuration. See the Django documentation for a complete list of available
parameters:
#   https://docs.djangoproject.com/en/stable/ref/settings/#databases
DATABASE = {
    'NAME': 'netbox',         # Database name
    'USER': 'netbox',               # PostgreSQL username
    'PASSWORD': '<redacted>',           # PostgreSQL password
    'HOST': 'netbox-postgres',      # Database server
    'PORT': '5432',               # Database port (leave blank for default)
    'CONN_MAX_AGE': 300,      # Max database connection age
}

# Redis database settings. Redis is used for caching and for queuing background tasks such as webhook events.
A separate
# configuration exists for each. Full connection details are required in both sections, and it is strongly
recommended
# to use two separate database IDs.
REDIS = {
    'tasks': {
        'HOST': 'netbox-redis',
        'PORT': 6379,
        # Comment out `HOST` and `PORT` lines and uncomment the following if using Redis Sentinel
        # 'SENTINELS': [('mysentinel.redis.example.com', 6379)],
        # 'SENTINEL_SERVICE': 'netbox',
        'PASSWORD': '',
        'DATABASE': 0,
        'SSL': False,
        # Set this to True to skip TLS certificate verification
        # This can expose the connection to attacks, be careful
        # 'INSECURE_SKIP_TLS_VERIFY': False,
    },
    'caching': {
        'HOST': 'netbox-redis',
        'PORT': 6379,
        # Comment out `HOST` and `PORT` lines and uncomment the following if using Redis Sentinel
        # 'SENTINELS': [('mysentinel.redis.example.com', 6379)],
        # 'SENTINEL_SERVICE': 'netbox',
        'PASSWORD': '',
        'DATABASE': 1,
```

```
        'SSL': False,
        # Set this to True to skip TLS certificate verification
        # This can expose the connection to attacks, be careful
        # 'INSECURE_SKIP_TLS_VERIFY': False,
    }
}

# This key is used for secure generation of random numbers and strings. It must never be exposed outside of this file.
# For optimal security, SECRET_KEY should be at least 50 characters in length and contain a mix of letters, numbers, and
# symbols. NetBox will not run without this defined. For more information, see
# https://docs.djangoproject.com/en/stable/ref/settings/#std:setting-SECRET_KEY
SECRET_KEY = '<redacted>'



#########################
#                       #
#   Optional settings   #
#                       #
#########################

# Specify one or more name and email address tuples representing NetBox administrators. These people will be notified of
# application errors (assuming correct email settings are provided).
ADMINS = [
    # ('John Doe', 'jdoe@example.com'),
]

# URL schemes that are allowed within links in NetBox
ALLOWED_URL_SCHEMES = (
    'file', 'ftp', 'ftps', 'http', 'https', 'irc', 'mailto', 'sftp', 'ssh', 'tel', 'telnet', 'tftp', 'vnc', 'xmpp',
)

# Optionally display a persistent banner at the top and/or bottom of every page. HTML is allowed. To display the same
# content in both banners, define BANNER_TOP and set BANNER_BOTTOM = BANNER_TOP.
BANNER_TOP = ''
BANNER_BOTTOM = ''
```

```
# Text to include on the login page above the login form. HTML is allowed.
BANNER_LOGIN = ''

# Base URL path if accessing NetBox within a directory. For example, if installed at https://example.com/netbox/,
set:
# BASE_PATH = 'netbox/'
BASE_PATH = ''

# Maximum number of days to retain logged changes. Set to 0 to retain changes indefinitely. (Default: 90)
CHANGELOG_RETENTION = 90

# API Cross-Origin Resource Sharing (CORS) settings. If CORS_ORIGIN_ALLOW_ALL is set to True, all origins will
be
# allowed. Otherwise, define a list of allowed origins using either CORS_ORIGIN_WHITELIST or
# CORS_ORIGIN_REGEX_WHITELIST. For more information, see https://github.com/ottoyiu/django-cors-headers
CORS_ORIGIN_ALLOW_ALL = False
CORS_ORIGIN_WHITELIST = [
    # 'https://hostname.example.com',
]
CORS_ORIGIN_REGEX_WHITELIST = [
    # r'^(https?://)?(\w+\.)?example\.com$',
]

# Specify any custom validators here, as a mapping of model to a list of validators classes. Validators should be
# instances of or inherit from CustomValidator.
# from extras.validators import CustomValidator
CUSTOM_VALIDATORS = {
    # 'dcim.site': [
    #     CustomValidator({
    #         'name': {
    #             'min_length': 10,
    #             'regex': r'\d{3}$',
    #         }
    #     })
    # ],
}

# Set to True to enable server debugging. WARNING: Debugging introduces a substantial performance penalty
and may reveal
# sensitive information about your installation. Only enable debugging while performing testing. Never enable
```

```
debugging
# on a production system.
DEBUG = False

# Email settings
EMAIL = {
    'SERVER': 'localhost',
    'PORT': 25,
    'USERNAME': '',
    'PASSWORD': '',
    'USE_SSL': False,
    'USE_TLS': False,
    'TIMEOUT': 10,  # seconds
    'FROM_EMAIL': '',
}

# Enforcement of unique IP space can be toggled on a per-VRF basis. To enforce unique IP space within the
global table
# (all prefixes and IP addresses not assigned to a VRF), set ENFORCE_GLOBAL_UNIQUE to True.
ENFORCE_GLOBAL_UNIQUE = False

# Exempt certain models from the enforcement of view permissions. Models listed here will be viewable by all
users and
# by anonymous users. List models in the form `<app>.<model>`. Add '*' to this list to exempt all models.
EXEMPT_VIEW_PERMISSIONS = [
    # 'dcim.site',
    # 'dcim.region',
    # 'ipam.prefix',
]

# Enable the GraphQL API
GRAPHQL_ENABLED = True

# HTTP proxies NetBox should use when sending outbound HTTP requests (e.g. for webhooks).
# HTTP_PROXIES = {
#     'http': 'http://10.10.1.10:3128',
#     'https': 'http://10.10.1.10:1080',
# }

# IP addresses recognized as internal to the system. The debugging toolbar will be available only to clients
```

```
accessing
# NetBox from an internal IP.
INTERNAL_IPS = ('127.0.0.1', '::1')

# Enable custom logging. Please see the Django documentation for detailed guidance on configuring custom
logs:
#   https://docs.djangoproject.com/en/stable/topics/logging/
LOGGING = {}

# Automatically reset the lifetime of a valid session upon each authenticated request. Enables users to remain
# authenticated to NetBox indefinitely.
LOGIN_PERSISTENCE = False

# Setting this to True will permit only authenticated users to access any part of NetBox. By default, anonymous
users
# are permitted to access most data in NetBox but not make any changes.
LOGIN_REQUIRED = False

# The length of time (in seconds) for which a user will remain logged into the web UI before being prompted to
# re-authenticate. (Default: 1209600 [14 days])
LOGIN_TIMEOUT = None

# Setting this to True will display a "maintenance mode" banner at the top of every page.
MAINTENANCE_MODE = False

# The URL to use when mapping physical addresses or GPS coordinates
MAPS_URL = 'https://maps.google.com/?q='

# An API consumer can request an arbitrary number of objects =by appending the "limit" parameter to the URL
(e.g.
# "?limit=1000"). This setting defines the maximum limit. Setting it to 0 or None will allow an API consumer to
request
# all objects by specifying "?limit=0".
MAX_PAGE_SIZE = 1000

# The file path where uploaded media such as image attachments are stored. A trailing slash is not needed.
Note that
# the default value of this setting is derived from the installed location.
# MEDIA_ROOT = '/opt/netbox/netbox/media'
```

```
# By default uploaded media is stored on the local filesystem. Using Django-storages is also supported. Provide the
# class path of the storage driver in STORAGE_BACKEND and any configuration options in STORAGE_CONFIG. For example:
# STORAGE_BACKEND = 'storages.backends.s3boto3.S3Boto3Storage'
# STORAGE_CONFIG = {
#     'AWS_ACCESS_KEY_ID': 'Key ID',
#     'AWS_SECRET_ACCESS_KEY': 'Secret',
#     'AWS_STORAGE_BUCKET_NAME': 'netbox',
#     'AWS_S3_REGION_NAME': 'eu-west-1',
# }

# Expose Prometheus monitoring metrics at the HTTP endpoint '/metrics'
METRICS_ENABLED = False

# Credentials that NetBox will uses to authenticate to devices when connecting via NAPALM.
NAPALM_USERNAME = ''
NAPALM_PASSWORD = ''

# NAPALM timeout (in seconds). (Default: 30)
NAPALM_TIMEOUT = 30

# NAPALM optional arguments (see https://napalm.readthedocs.io/en/latest/support/#optional-arguments). Arguments must
# be provided as a dictionary.
NAPALM_ARGS = {}

# Determine how many objects to display per page within a list. (Default: 50)
PAGINATE_COUNT = 50

# Enable installed plugins. Add the name of each plugin to the list.
PLUGINS = []

# Plugins configuration settings. These settings are used by various plugins that the user may have installed.
# Each key in the dictionary is the name of an installed plugin and its value is a dictionary of settings.
# PLUGINS_CONFIG = {
#     'my_plugin': {
#         'foo': 'bar',
#         'buzz': 'bazz'
#     }
```

```
# }

# When determining the primary IP address for a device, IPv6 is preferred over IPv4 by default. Set this to True to
# prefer IPv4 instead.
PREFER_IPV4 = False

# Rack elevation size defaults, in pixels. For best results, the ratio of width to height should be roughly 10:1.
RACK_ELEVATION_DEFAULT_UNIT_HEIGHT = 22
RACK_ELEVATION_DEFAULT_UNIT_WIDTH = 220

# Remote authentication support
REMOTE_AUTH_ENABLED = False
REMOTE_AUTH_BACKEND = 'netbox.authentication.RemoteUserBackend'
REMOTE_AUTH_HEADER = 'HTTP_REMOTE_USER'
REMOTE_AUTH_AUTO_CREATE_USER = False
REMOTE_AUTH_DEFAULT_GROUPS = []
REMOTE_AUTH_DEFAULT_PERMISSIONS = {}

# This repository is used to check whether there is a new release of NetBox available. Set to None to disable the
# version check or use the URL below to check for release in the official NetBox repository.
RELEASE_CHECK_URL = None
# RELEASE_CHECK_URL = 'https://api.github.com/repos/netbox-community/netbox/releases'

# The file path where custom reports will be stored. A trailing slash is not needed. Note that the default value of
# this setting is derived from the installed location.
# REPORTS_ROOT = '/opt/netbox/netbox/reports'

# Maximum execution time for background tasks, in seconds.
RQ_DEFAULT_TIMEOUT = 300

# The file path where custom scripts will be stored. A trailing slash is not needed. Note that the default value of
# this setting is derived from the installed location.
SCRIPTS_ROOT = '/config/scripts'

# The name to use for the session cookie.
SESSION_COOKIE_NAME = 'sessionid'

# By default, NetBox will store session data in the database. Alternatively, a file path can be specified here to use
```

```
# local file storage instead. (This can be useful for enabling authentication on a standby instance with read-only
# database access.) Note that the user as which NetBox runs must have read and write permissions to this path.
SESSION_FILE_PATH = None

# Time zone (default: UTC)
TIME_ZONE = 'UTC'

# Date/time formatting. See the following link for supported formats:
# https://docs.djangoproject.com/en/stable/ref/templates/builtins/#date
DATE_FORMAT = 'N j, Y'
SHORT_DATE_FORMAT = 'Y-m-d'
TIME_FORMAT = 'g:i a'
SHORT_TIME_FORMAT = 'H:i:s'
DATETIME_FORMAT = 'N j, Y g:i a'
SHORT_DATETIME_FORMAT = 'Y-m-d H:i'
```

_Docker Daemon

These configs have stumped me one too many times.

Issue: Why is my docker host having trouble resolving DNS?

After enabling pi-hole or other dockerized DNS management app, lack of (systemd-resolved e.t.c. means that DNS is not available on a local level prior to docker to so much as pull pi-hole. (wonderful chick needs to lay its own egg scenario) A great solution I found to this issue is to set the DNS for docker explicitly.

Issue: Why aren't the nvidia drivers workable on my system.

Gotta love the oversight, better remember to request gpu here as well as in the compose. 🤦‍♀️

Issue: After creating about 40 stacks / docker containers ... I can no longer create new ones with unique ips...

As seen below we need to set/allow additional ranges of IP addresses for docker to use in addition to those available by default. You can use something like below or adjust to a set of IPs not taken elsewhere on your network.

```
{
   "runtimes": {
      "nvidia": {
         "args": [],
         "path": "nvidia-container-runtime"
      }
   },

   "default-address-pools" : [
      {
         "base" : "172.50.0.0/12",
         "size" : 20
         },
      {
         "base" : "192.168.30.0/16",
         "size" : 24
      }
   ],
   "dns": ["<pihole#1>", "<pihole#2>", "1.1.1.1", "8.8.8.8"]
}
```

Note:
Later when I describe how to setup pihole remind me to note that we need to turn off that below

and update the dns for docker.

Pi-Hole
sudo systemctl stop systemd-resolved